0 目录

1 概述

尊敬的用户：

您好！非常感谢您使用伟福系列仿真器。伟福仿真品种多、功能强，和国内外同类高档仿真器功能相比，软、硬件方面具有多种先进特点。

硬件方面先进的特点：

1. **通用仿真器**：主机+POD 组合，通过更换 POD，可以对各种 CPU 进行仿真。对不同的应用场合，用户如果选择不同的 CPU，通常就要更换仿真器，而伟福仿真器则采用主机+POD 组合，支持多类 CPU 仿真。只需通过更换不同的 POD，即可对各种不同类型的单片机进行仿真。为用户提供了一种灵活的多 CPU 仿真系统。
2. **仿真 CPU 外置**：直接位于用户板的上方，提高仿真频率以及降低信号噪声，而无须缩短您的仿真电缆。
3. **强大的逻辑分析仪综合调试功能**：逻辑分析仪由交互式软件菜单窗口对系统硬件的逻辑或进序进行同步实时采样，并实时在线调试分析，采集深度 32K(E6000/L)，最高时基采样频率达 20M，40 路波形的可精确实时反映用户程序运行时的历史时间。系统在使用逻辑分析仪时，除普通的单步运行、键盘断点运行、全速硬件断点运行外，还可实现各种条件组合断点如：数据、地址、外部控制信号、CPU 内部控制信号、程序区间断点等。由于逻辑分析仪可以直接对程序的执行结果进行分析，因此极大地便利于程序的调试。随着科学技术的发展，单片机通讯方面的运用越来越多。在通讯功能的调试时，如果通讯不正常，查找原因是非常耗时和低效的，您很难搞清楚问题到底在什么地方，是波特率不对，是硬件信道有问题，是通讯协议有问题，是发方出错还是收方出错。有了逻辑分析仪，情况则完全不一样，用它可以分别或者同时对发送方、接收方的输入或者输出波形进行记录、存储、对比、测量等各种直观的分析，可以将实际输出通讯报文的波形与源程序相比较，可立即发现问题所在，从而极大地方便了调试。
4. **强大的跟踪器功能**：跟踪功能以总线周期为单位，实时记录 CPU 仿真运行过程中，总线上发生的事件，其触发条件方式同逻辑分析仪。跟踪窗口在仿真停止时可收集显示跟踪的 CPU 指令记忆信息，可以以总线反汇编码模式、源程序模式对应显示跟踪结果。屏幕窗口显示波形图最多跟踪记忆指令 32K 并通过仿真器的断点、单步、全速运行或各种条件组合断点来完成跟踪功能。总线跟踪可以跟踪程序的运行轨迹。可以统计软件运行时间。
5. **波形发生器功能**：伟福 E6000/L 仿真器可以输出 8 路可编程数字波形，波形深度达 32K，最高频率为 20MHz。用户在设计初期和测试时常常为没有理想的输出信号源而苦恼，一些

简单的脉冲又不能满足逻辑时序的要求。这时就非常需要有一种波形宽度可编程，相互时序可编程，波形的产生又能与用户的程序运行同步，脉冲波形频率可选择的设备。伟福仿真器 E6000/L 所附带的波形发生器就能满足您的要求，它可以向用户板上输出多达 8 路可编程的与程序同步的复杂数字波形，为设计人员提供各种数字信号源，例如常用的串行通信信号，I2C，SPI 等波形。

6. **影子存储器**：用户在程序全速执行时，可以实时观察到时 MCS51 系列 CPU 和 MCS96 系列 CPU 的外部数据的变化。用户常常希望在程序全速执行时，外部数据的变化情况，以及时了解程序的运行状态，影子存储器可以在程序运行时，为外部存储器建立映像，从而可以实时观察到外部数据的变化。
7. **代码覆盖**：使用此功能可观察到源程序各代码行，是否被执行过。在程序执行多分支结构的程序，用户常常希望知道那些代码被执行，那些代码没有被执行。代码覆盖功能在源程序行以不同的颜色标志程序执行情况。
8. **程序时效分析**：统计每个函数、过程运行时间，以及占整个程序运行时间的百分比。在设计高效率程序时，就要知道程序中各函数、各过程运行时间及占总时间的百分比，程序时效分析可以对此进行统计分析。
9. **数据时效分析**：与程序时效分析相似的是，数据时效分析，它可统计每个变量被访问的次数及占整个程序访问次数的百分比。通过数据时效分析可以数据的访问效率，以优化存储器单元配置。
10. **硬件测试**：对于 MCS51 系列 CPU 和 MCS96 系列 CPU 可以静态地输出地址、数据以及 ALE、PSEN、BHE、RD、WR 等读写控制信号，从而可以从用户板上静态地测量这些信号的值，从底层去控制、分析电路的工作状态，可以准确方便地检测硬件方面的隐蔽问题。
11. **事件触发**：用于指定用户程序运行时，出现的各种事件，这些事件包括地址条件、数据条件、控制信号条件、外部信号条件以及这些条件的组合，用这些事件来触发、控制逻辑分析仪、程序跟踪器的运行，以捕捉程序运行时出现的各类复杂情况，迅速定位设计中软、硬件问题所在。
12. **记时器**：记录程序运行时间。
13. **双 CPU 结构**: 由监控 CPU 控制仿真 CPU 完成仿真工作，100% 不占用户资源。全空间硬件断点，不受任何条件限制，支持地址、数据、外部信号、事件断点、支持实时断点计数、软件运行时间统计。

软件方面先进的特点：

14. **Wave6000 及 Keil uVision 双平台**：Wave6000 IDE 环境，中/英文界面可任选，用户源程序的大小不再有任何限制。有丰富的窗口显示方式，多方位，动态地展示仿真的各种过程，使用极为便利。仿真器同时还可以直接工作于 Keil uVision 调试环境下，适应不同的用户操作习惯。
15. **双工作模式**：a）软件模拟仿真（不用仿真器也能模拟运行用户程序）。b）硬件仿真。
16. **真正集成调试环境**：集成了编辑器、编译器、调试器，源程序编辑、编译、下载、调试全部可以在一个环境下完成。且伟福的多种仿真器，及所支持各种 CPU 仿真全部集成在一个环境下。可仿真 MCS51 系列，MCS196 系列，Microchip PIC 系列 CPU。为了跟上形势，现在很多工程师需要面对和掌握不同和项目管理器、编辑器、编译器。他们由不同的厂家开发，相互不兼容，使用不同的界面。学习使用都很吃力。伟福 WINDOWS 调试软件为您提供了一个全集成环境，统一的界面，包含一个项目管理器，一个功能强大的编辑器，汇编 Make、Build 和调试工具并提供一个与第三方编译器的接口。由于风格统一，从而大大节省了您的精力和时间。
17. **项目管理功能**：现在单片机软件越来越大，也越来越复杂，维护成本也很高，通过项目管理可化大为小，化繁为简，便于管理。项目管理功能 也使得多模块，多语言混合编程成为可能。
18. **多语言多模块混合调试**：支持 ASM（汇编）、PLM、C 语言多模块混合源程序调试，在线直接修改、编译、调试源程序。如果源程序有错，可直接定位错误所在行。
19. **直接点屏观察变量**：在源程序窗口，点击变量就可以观察此变量的值，方便快捷。
20. **功能强大的变量观察**：支持 C 语言的复杂类型，树状结构显示变量，
21. **强大的书签、断点管理功能**：书签、断点功能可快速定位程序，为编写、查找、比较程序提供帮助。
22. **类似 IE 的前进、后退定位功能**：可以在项目内跨模块地定位光标前一次或后一次位置，为比较、分析程序提供帮助。
23. **类似 Delphi 的界面操作**：类似 Delphi 的集成调试环境，灵活多变的窗口“靠岸”(Docking) 功能，可以方便地将窗口平排靠岸，或以页面方式靠岸，任由用户自己按排。桌面整洁，操作灵活。
24. **方便实用、功能多样的源程序编辑窗口**：a.）窗口分隔功能可将源程序窗口分成两个完全独立的编辑窗口，而所编辑的内容却是同一程序，为分析、比较检查大程序提供方便。b.）语法相关彩色显示，使得编写程序轻松，观察程序醒目。且用户可自己定义所喜好颜色，享受个性化编程带来的快乐 c.）书签功能提供多达 9 个书签，使得您在分析、比较、检查大程序时从容不迫。d.）寻找配对符号功能为您在复杂程序嵌套中找到“另一半”，例如可以找与‘{’相对的‘}’，或为‘(’找到相对的‘)’。e.）多行程序的同进同退功能，可以使得程序错落有致，帮您编写优美、整洁的程序。

25. **外设管理功能**：外设管理可以让您在调试程序时，观察到端口、定时器、串行口中断、外部中断相关的寄存器的状态，更可以帮您完成这些外设的初始化程序，包括 C 语言和汇编语言，而您所做的只是填表，定义外设所要完成的功能。
26. **功能独特的反汇编功能**： 伟福独创的控制文件方式的反汇编功能，可以帮助你将机器码反汇编成工整的汇编语言，通过控制文件你可以定义程序中数据区、程序区、无用数据区，还可将一些数据、地址定义成符号，便于阅读。你若丢了你的源程序，它可帮你迅速恢复。

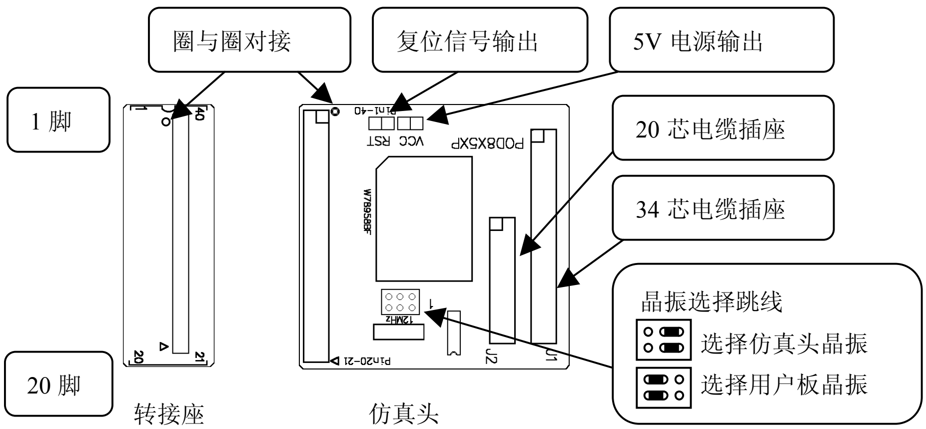
南京伟福实业有限公司

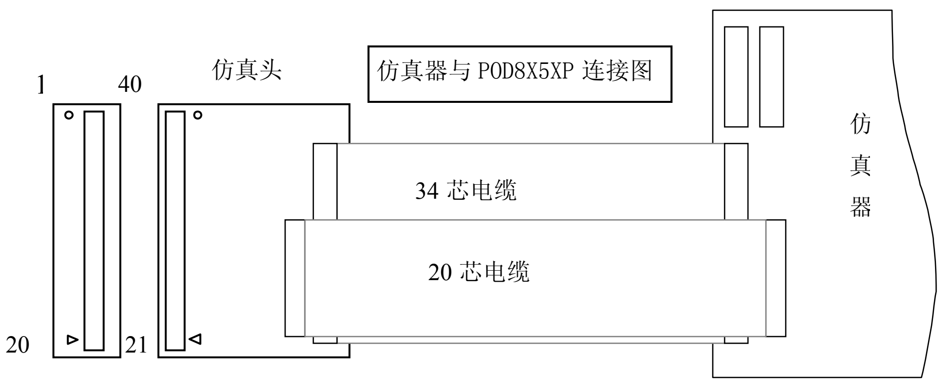
电话：（025）3193973，3192459
传真：（025）3192459
电邮：wave-cn@263.net
网址：http://www.wave-cn.com

2 仿真器硬件

一. 仿真头介绍

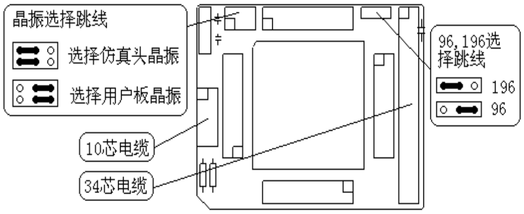
◆ POD8X5XP 仿真头



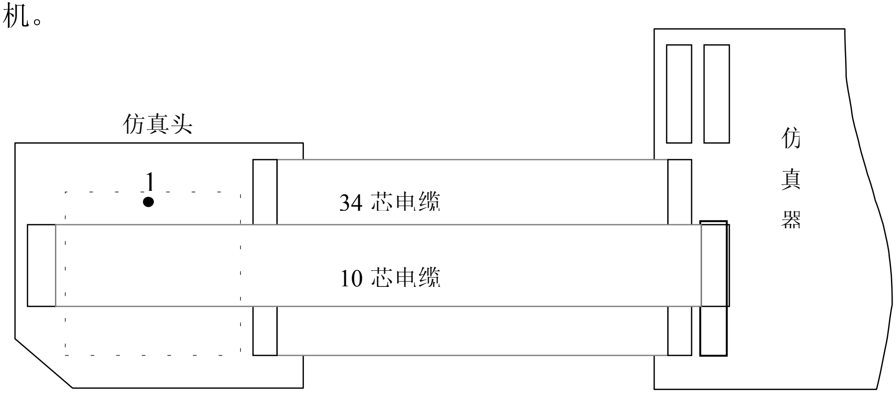


POD8X5XP 仿真头为 POD8X5X 改进型。可配 E2000 系列,E6000 系列,K51 系列仿真器,用于仿真 MCS51 系列及兼容单片机,可仿真 CPU 种类为 8031/32, 8051/52, 875X, 89C5X, 89CX051, 华邦的 78E5X, LG 的 97C51/52/1051/2051。 配有 40 脚 DIP 封装的转接座,可选配 44 脚 PLCC 封装的转接座.选配 2051 转接座可仿真 20 脚 DIP 封装的 89CX051CPU。当用户板功耗不大时，可以短接 5V 电源输出跳线，由仿真器供电给用户板，一般情况下请不要短接此跳线。如果短接复位信号输出跳线，当用软件复位程序时，仿真头的复位脚会输出一个复位信号，以复位用户板的其它器件。注意：如果用户板有复位电路，请不要短接此跳线。

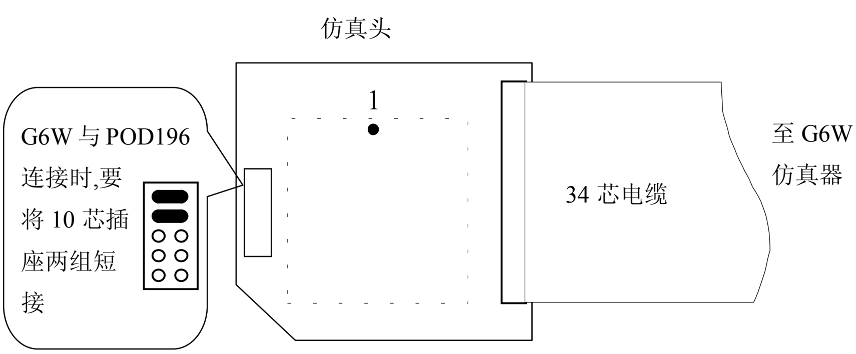
◆ POD196KB/KC 仿真头



用于仿真 INTEL80C196KB/KC 单片机，若将仿真头的 196KC 换成 196KD，还可以仿真 80C196KD 单片机。

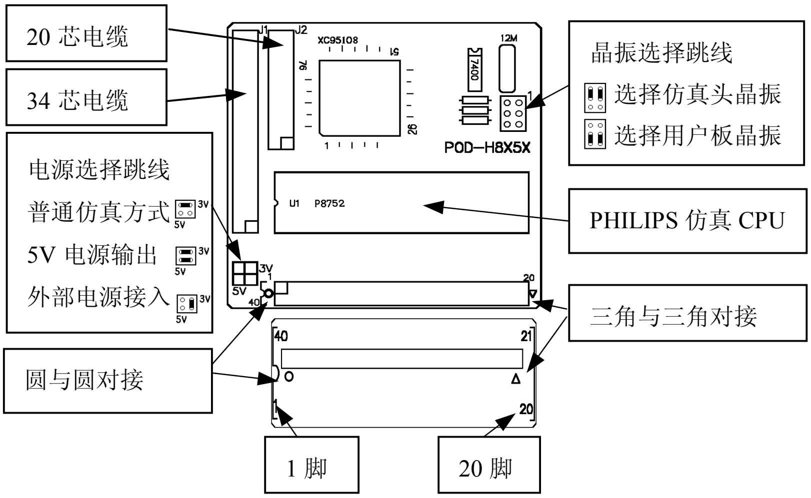


仿真器与 POD196KC 连接图



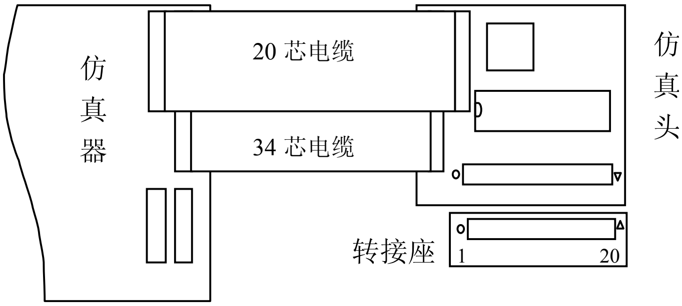
G6W 仿真器与 POD196KC 连接图

◆ PODH8X5X / PODH591 仿真头



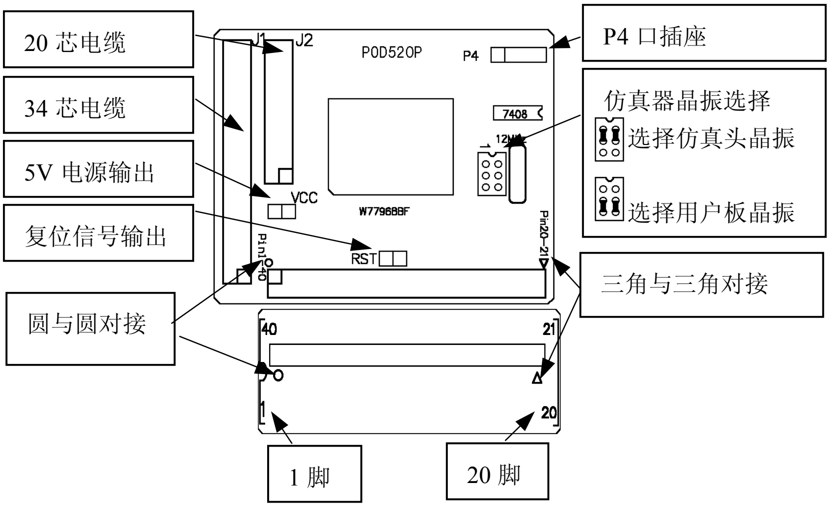
PODH8X5X 运用 PHILIPS 授权的 HOOKS 技术，用 PHILIPS 芯片作为仿真芯片，来仿真各类与 MCS51 兼容的 MCU，仿真头的原有的 P87C52 可仿真通用的 8X5X 系列芯片，可以将 P87C52 换成 PHILIPS 的 P89C51Rx+或 P89C51Rx2 来仿真相应的 MCU，也可以换成 PHILIPS 的 P89C66x 用于仿真 PHILIPS 的 P89C66x 系列 MCU。因为 P89C51RD2 和 P89C66X 内部带有扩展 RAM，可以借用 P89C51RD2 或 P89C66x 来仿真带扩展 RAM 的 CPU，例如 Winbond 的 78E58B、78E516 等。

PODH8X5X 可以从外部引入仿真电源，来仿真 2.7V~5.5V 用户电压，当用户需要仿真低电压时，将“电源选择跳线”接成“外部电源接入”方式即可。仿真头的低电压由用户板提供。注意：当用户想仿真低电压时，仿真头上的仿真 CPU 必须能工作于低电压状态。（详见 PODH8X5X 使用说明）

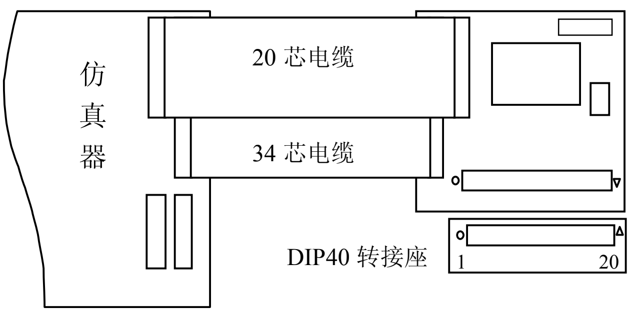


仿真器与 PODH8X5X 仿真头连接图

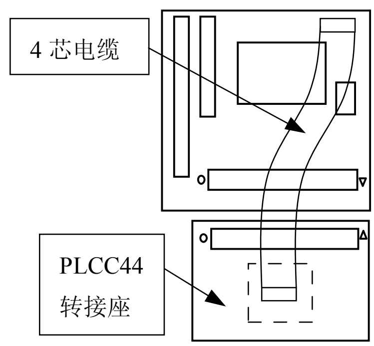
◆ POD520P 仿真头



用于仿真 DALLAS 的 80C320,80C520,87C520,WINBOND 的 77E58 高速单片机,配有 40 脚 DIP 封装的转接座,可选配 44 脚 PLCC 封装的转接座。当用户板功耗不大时，可以短接 5V 电源输出跳线，由仿真器供电给用户板，一般情况下请不要短接此跳线。如果短接复位信号输出跳线，当用软件复位程序时，仿真头的复位脚会输出一个复位信号，以复位用户板的其它器件。特别要注意：如果用户板有复位电路，请不要短接此跳线。当用户使用 PLCC44 封装的芯片时，将仿真头上的 P4 口四芯插座和 PLCC44 转接座的四芯插座连接，就可以仿真 77E58 的 P4 口功能。

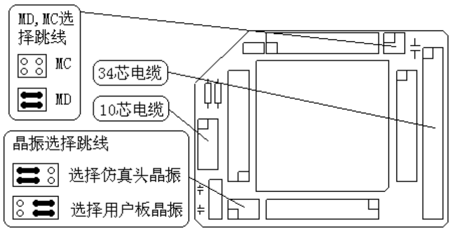


E6000 仿真器与 POD520P 连接图

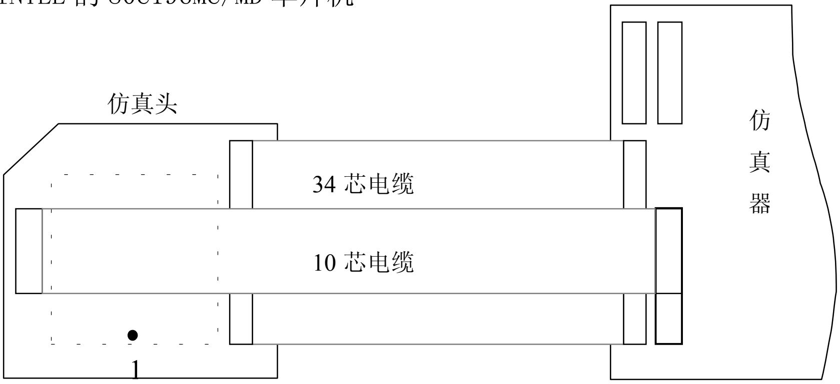


PLCC44 转接与仿真头的连接

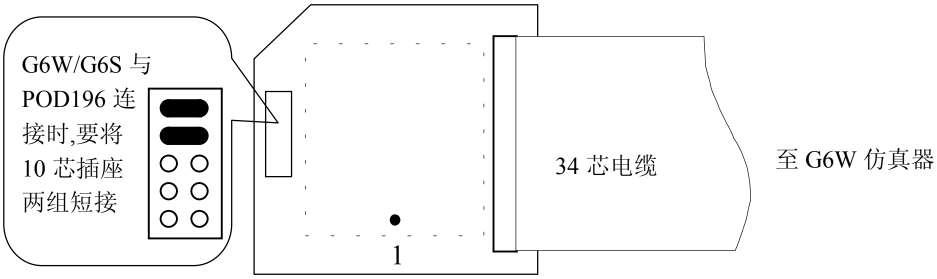
◆ POD196MC/MD 仿真头



用于仿真 INTEL 的 80C196MC/MD 单片机



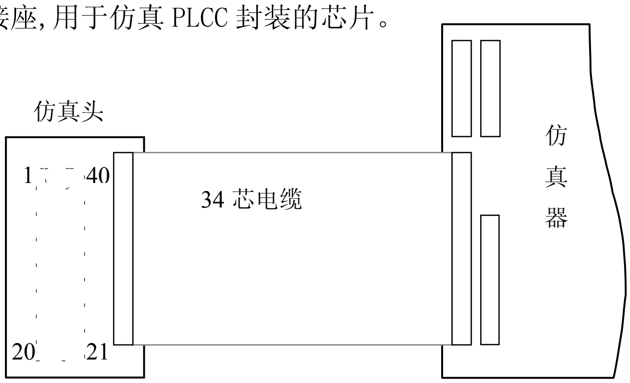
E6000 仿真器与 POD196MC 连接图

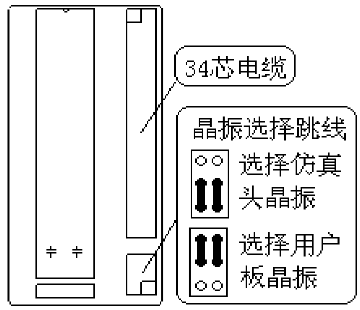


G6W 仿真器与 POD196MC 连接图

◆ POD8051 仿真头

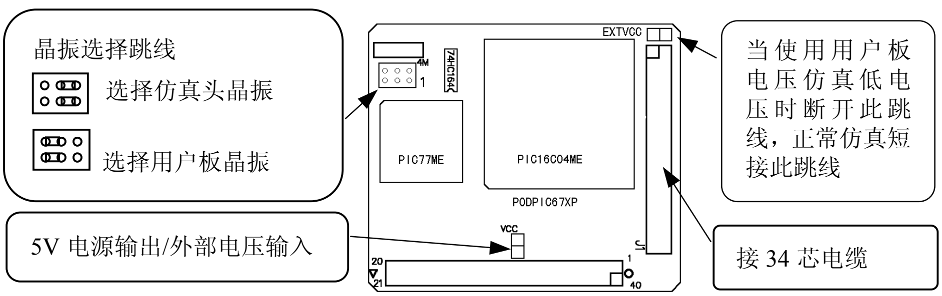
用于仿真 P0,P2 口做为总线工作方式的8031/32,8051/52系列及兼容单片机,可选配44脚PLCC封装的转接座,用于仿真PLCC封装的芯片。





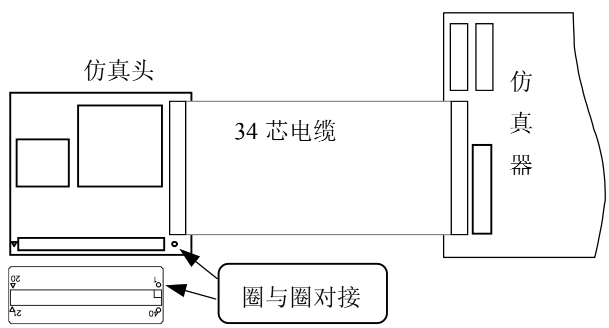
E6000 / G6W 仿真器与 POD8051 连接图

◆ PODPIC67XP 仿真头

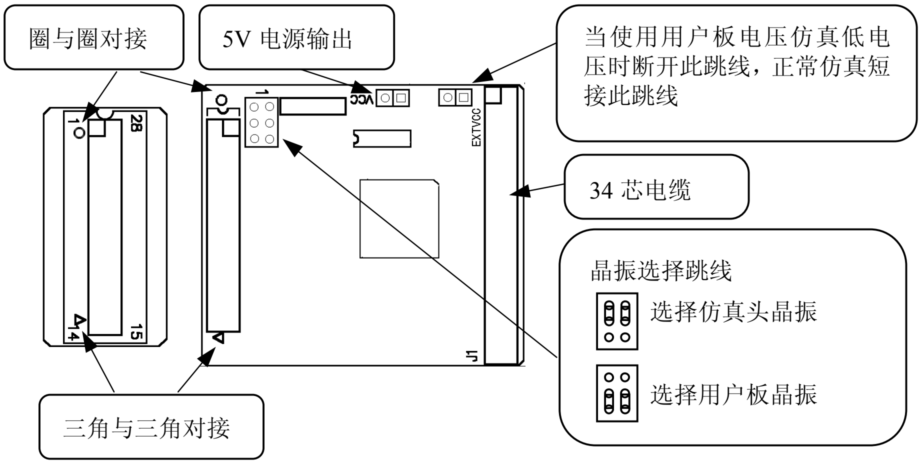


PODPIC67XP 为 POD16C6X/7X 的改进型，用于仿真 Microchip PIC16C6x/7x 单片机,可仿真芯片与原 POD16C6X/7X 相同。1）**正常仿真**时，短接 EXTVCC 跳线，仿真头电压由仿真器提供。2）如果用户需要**仿真低电压**，请断开 EXTVCC 跳线，同时短接 VCC 跳线，这时，低电压由用户提供。3）正常仿真时 VCC 跳线也可当做**电压输出**用，当用户板功耗不大时，可以短接 5V 电源输出跳线，由仿真器供电给用户板，一般情况下请不要短接此跳线。因为 PIC711 系列 CPU 与 PIC72 以后芯片寄存器有所不同,所以在用 PODPIC67XP 仿真 PIC711 时，应该注意寄存器的定义。（**见如何用 PODPIC67XP 仿真 PIC711**）

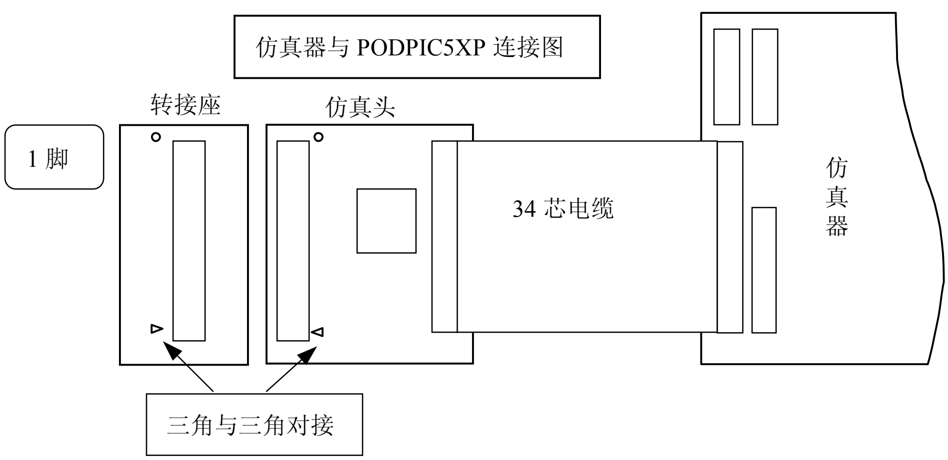
仿真器与 PODPIC67XP 连接图



◆ PODPIC5XP 仿真头

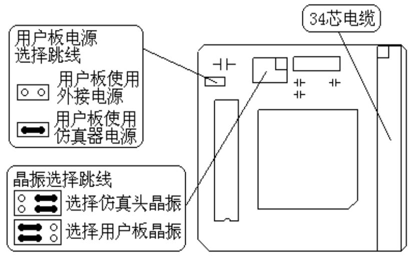


PODPIC5XP 为 PODPIC5X 的改进型，用于仿真 Microchip 公司的 PIC16C5X 系列单片机，可仿真的芯片与 POD16C5X 相同。1）**正常仿真**时，短接 EXTVCC 跳线，由仿真器供电给仿真头。2）如果用户需要**仿真低电压**，请断开 EXTVCC 跳线，同时短接 VCC 跳线，这时，仿真头的低电压由用户提供。3）正常仿真时（EXTVCC 跳线短接），VCC 跳线也可当做**电压输出**用，当用户板功耗不大时，可以短接 5V 电源输出跳线，由仿真器供电给用户板，注意：此时用户板不能再外接电源。一般情况下请不要短接此跳线。

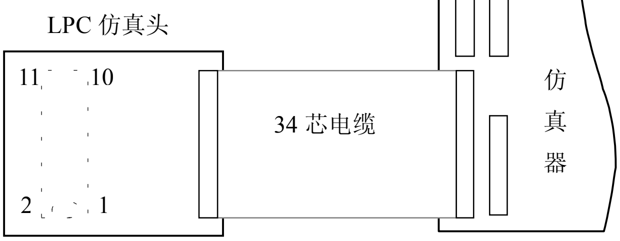


◆ PODLPC76X 仿真头

用于仿真 PHILIPS 的 LPC76X 系列单片机，若仿真芯片为 LPC769，那么可仿真 PHILIPS 公司的 87LPC759/760/761/762/764/767/**769** 系列单片机。若仿真芯片为 LPC768，那么可仿真的芯片为 87LPC759/760/761/762/764/767/**768**。



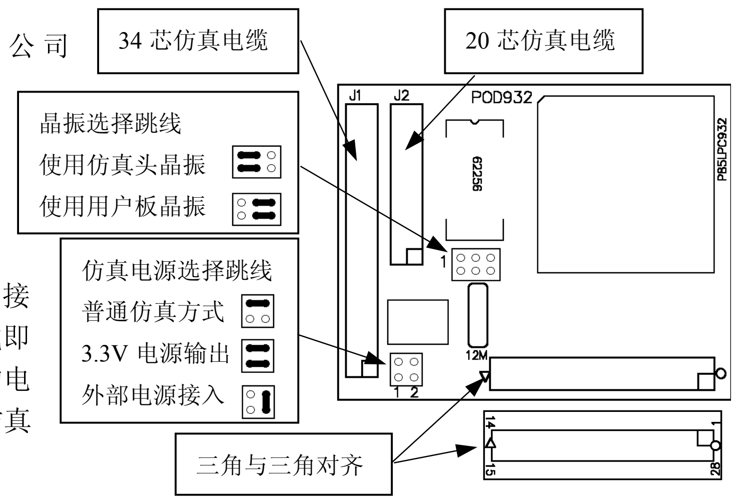
仿真器与 PODLPC 连接图



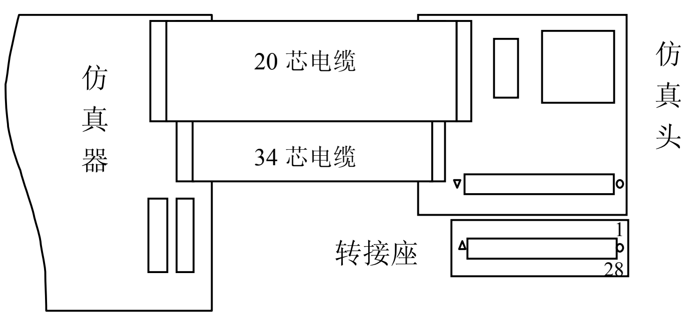
◆ PODLPC93X 仿真头

用于仿真 Philips 公司 LPC93x 系列芯片。

PODLPC93X 可以从外部引入仿真电源，来仿真 2.7V 的用户电压，当用户需要仿真低电压时，将“电源选择跳线”接成“外部电源接入”方式即可。注意：此时用户板的电压请勿过高，以免烧坏仿真头。

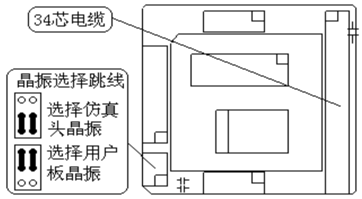


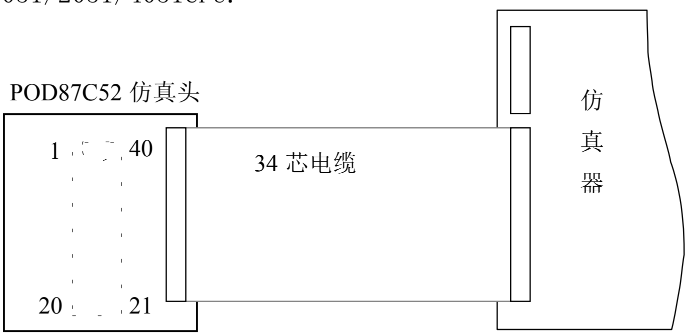
E6000 仿真器与 PODLPC93X 连接图。（PODLPC93X 与 LPC93X 专用仿真器连接时，只需接 34 芯电缆）



◆ POD87C52 仿真头

POD87C52 仿真头用于仿真 P0,P2 做为 I/O 口工作方式的 89C51/52,87C51/52 系列单片机,此种仿真头适用于 G6W 型仿真器. 可选配 44 脚 PLCC 封装的转接座,用于仿真 PLCC 封装的芯片. 选配 2051 转接可仿真 20 脚 DIP 封装的 XXC1051/2051/4051CPU.

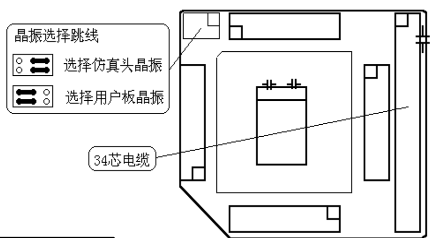


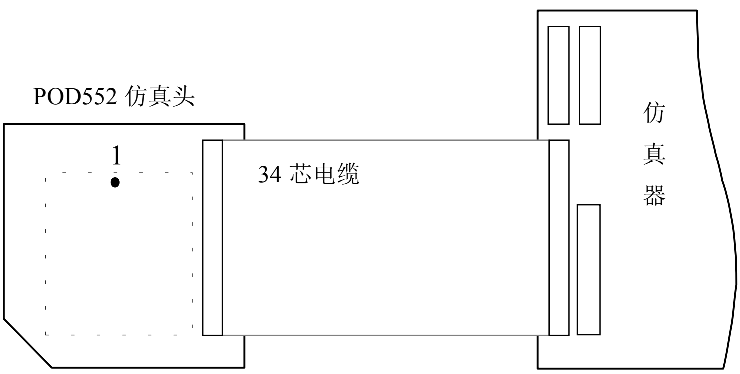


G6W 仿真器与 POD87C52 连接图

◆ POD552 / POD592 仿真头

POD552 仿真头用于仿真 PHILIPS 公司的 80C552 单片机，P0/P2 口工作于总线方式，POD592 仿真头用于仿真 PHILIPS 公司的带 CAN 总线的 80C592 仿真头，P0/P2 口工作于总线方式。

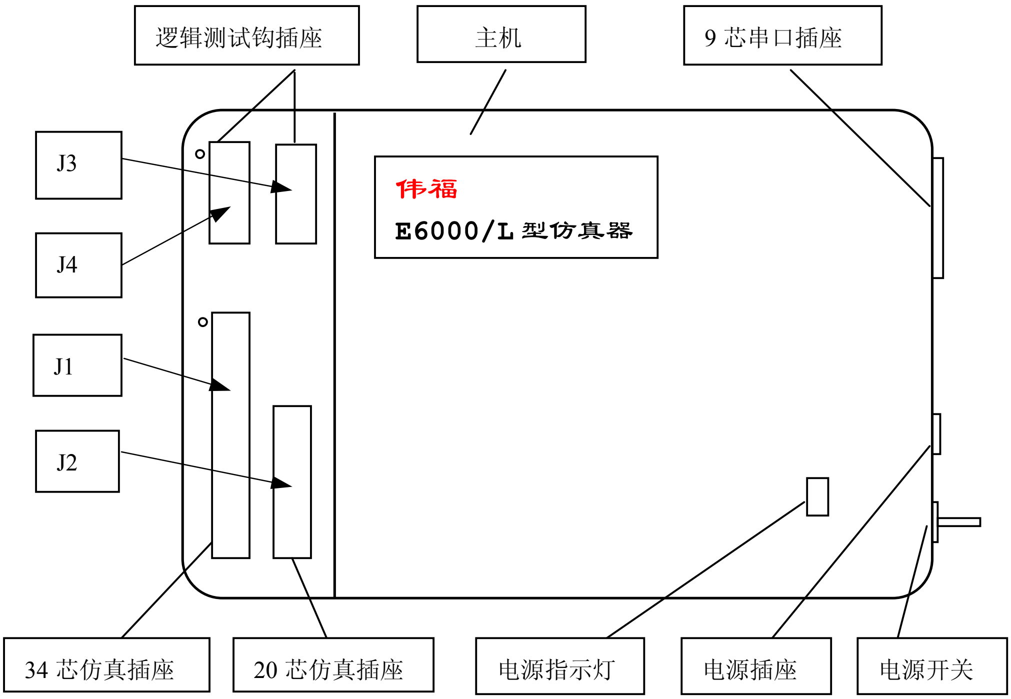




仿真器与 POD552 连接图

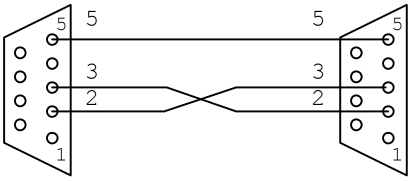
二. 仿真器介绍

外形示意图



说明：

1. 仿真器使用 9 针串行口，与 PC 机用两头为孔的串行电缆连接。对于一些只有 USB 口而没有串口的计算机，可以使用 USB 转串口电缆将 USB 转成串行口。串行电缆内部连接为：



2. 根据仿真器型号不同，逻辑测试钩插座可能只有一个。
3. 根据仿真器型号不同，可能会没有 20 芯仿真电缆插座。
4. 电源为直流 5V/1A（最小），电源插孔的极性为内“正”外“负”。

E6000L/E6000T/E6000S 型仿真器

仿真器型号	功能
E6000/S	通用仿真器（1-16 位，15M 总线速度） 硬件测试仪 运行时间统计 逻辑笔（选配件） WINDOWS 版本、DOS 版本双平台、支持 Keil uVision 开发环境
E6000/T	含 E6000/S 所有功能 事件断点、断点记数 跟踪器 影子存储器 全空间程序/数据时效分析器
E6000/L	含 E6000/T 所有功能 逻辑分析仪（测试钩为选配件) 可编程波形发生器

E6000 系列仿真可配置仿真头

仿真头型号	可仿真 CPU
POD8X5XP	8X5X 系列（P0 口和 P2 口作为总线或 IO 口用）
PODH8X5X	Philips 40 脚及 44 脚 51 系列芯片及通用 8X5X 芯片
POD520P	Dalas310/320/510/520/华邦 77E58
POD51	8X5X 系列 CPU（P0 口和 P2 口作为总线用）
POD2051	2051、1051 系列 CPU（需与 POD8X5XP 配合使用）
POD552	Philips 80C552（P0 口和 P2 口做为总线）
POD592	Philips 80C592（P0 口和 P2 口做为总线）
PODLPC93X	Philips LPC93X
PODLPC76X	Philips LPC76X
POD196KC	196KC/KB/KD
POD196MC	196MC/MD/MH
POD16C5XP	PIC16C52/54/55/56/57/58、PIC12C508/509
POD16C67XP	PIC16C61/62/63/64/65/67、PIC16C71/72/73/74/76/77
POD16C71X	PIC16C71X 全系列
POD16C8X	PIC16C83/84
PODH591	Philips P87C591 芯片
新仿真头	伟福公司将为 E6000 系列开发新的仿真头

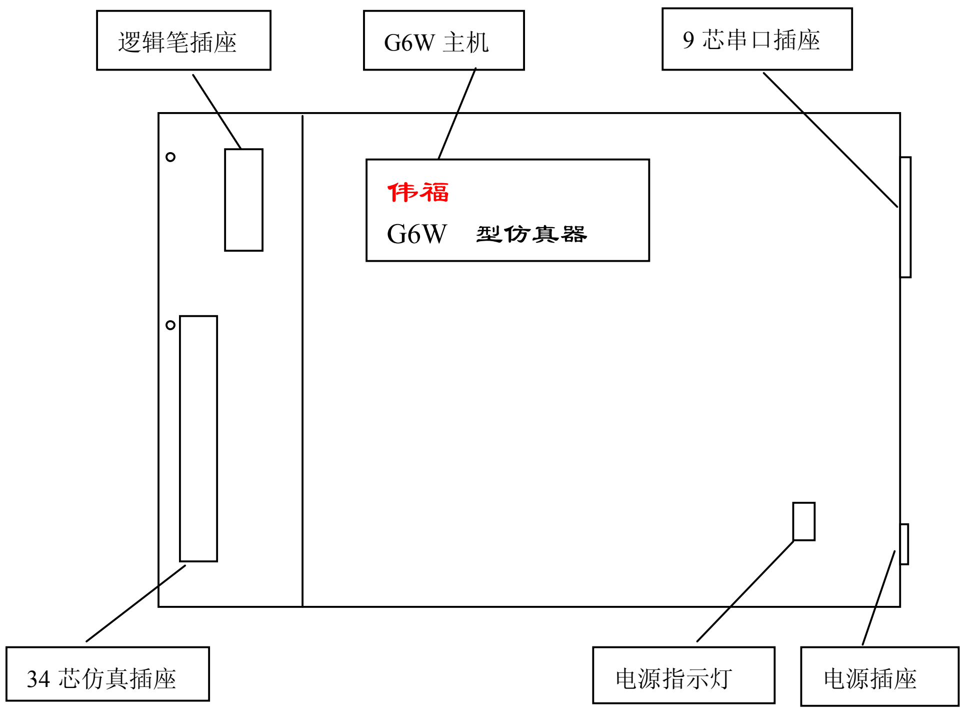
G6W 型仿真器

仿真器型号	功能
G6W	通用仿真器 （1-16 位，10M 总线速度） 硬件测试仪 WINDOWS 版本、DOS 版本

G6W 可配置仿真头

仿真头型号	可仿真 CPU
POD87C52	8X5X 系列（P0 口和 P2 口作为 IO 口用）
POD51	8X5X 系列 CPU（P0 口和 P2 口作为总线用）
POD2051	2051、1051 系列 CPU（需与 POD8752 配合使用）
POD552	Philips 80C552（P0 口和 P2 口作为总线）
POD196KC	Intel 80C196KC/KB
POD196MC	Intel 80C196MC/MD

G6W 外形示意图



K51L/K51T/K51S 51 系列专用型仿真器

仿真器型号	功能	可配仿真头
K51/S	51 系列专用仿真器（Bondout 仿真技术） 运行时间统计 最高仿真频率可达 40MHz 逻辑笔（选配件） WINDOWS 版本、DOS 版本双平台、Keil uV2 环境	POD8X5XP，用于仿真 8X5X 系列单片机，（P0 口和 P2 口可做总线和 I/O 口用）
K51/T	含 K51/S 所有功能 跟踪器	
K51/L	含 K51/T 所有功能 逻辑分析仪(外接 8 路,逻辑探钩为选配件)	

Philips H51L/H51T/H51S 系列专用型仿真器

仿真器型号	功能	可配仿真头
H51/S	PhilipsHOOKS 专用仿真器（Hooks 仿真技术） 2.7V 至 5.5V 宽电压 0 至 24MHz 宽频率 WAVE6000 及 Keil uVision 双平台	PODH8X5X,用于仿真通用的 8X5X 芯片及 Philips 的 40 脚及 44 脚 51 指令集芯片 PODH591(选配),用于仿真 Philips 的 87C591 芯片。
H51/T	含 H51/S 所有功能 跟踪器	
H51/L	含 H51/T 所有功能 逻辑分析仪(外接 8 路,逻辑探钩为选配件)	

Philips LPC76X 专用型仿真器

仿真器型号	功能	可配仿真头
LPC76X	LPC76X 系列专用仿真器（Bondout 仿真技术）。 2.7V 至 5.5V 宽电压 0 至 20MHz 宽频率 WINDOWS 版本软件	PODLPC76X，用于仿真 Philips 87C76X 系列单片机。

Philips LPC93X 专用型仿真器

仿真器型号	功能	可配仿真头
LPC93X	LPC93X 系列专用仿真器(Bondout 仿真技术) 2.7V 至 3.6V 宽电压 0 至 12MHz 宽频率 WINDOWS 版本软件	PODLPC93X，用于仿真 Philips 89C93X 系列单片机

Philips P51 系列 Philips 通用型仿真器

仿真器型号	功能	可配仿真头
P51	Philips 通用仿真器（Hooks 及 Bondout 仿真技术） 2.7V 至 5.5V 宽电压 0 至 20MHz 宽频率 WAVE6000 及 Keil uVision 双平台	PODH8X5X， POD591（选配） POD76X（选配） POD93X（选配）

MicroChip PIC6000 系列专用型仿真器

仿真器型号	功能	可配仿真头
PIC6000	Microchip 专用仿真器（Bondout 仿真技术） 2.7V 至 5.5V 宽电压 0 至 12MHz 宽频率 WAVE6000 WINDOWS 版本软件	PODPIC5XP（选配） PODPIC67XP（选配）

3 软件安装

WINDOWS 版本软件安装

1.将光盘放入光驱，光盘会自动运行，出现安装提示。
2.选择“安装 WINDOWS”软件
3.按照安装程序的提示,输入相应内容。
4.继续安装,直至结束.

若光驱自动运行被关闭，用户可以打开光盘的\ICESSOFT\E2000W\目录（文件夹），执行 SETUP.EXE，按照安装程序的提示，输入相应的内容，直至结束。

在安装过程中,如果用户没有指定安装目录,安装完成后,会在 C:盘建立一个 C:\WAVE6000 目录(文件夹),结构如下:

目录	内容
C:\WAVE6000	
├ BIN	可执行程序及相关配置文件
├ HELP	帮助文件和使用说明
└ SAMPLES	样例和演示程序

可以从公司网站下载软件的最新版本。网站网址为 http://www.wave-cn.com

编译器安装

伟福仿真系统已内嵌汇编编译器(伟福汇编器),同时留有第三方的编译器的接口,方便用户使用高级语言调试程序.编译器请用户自备.

安装 51 系列 CPU 的编译器

1.进入 C:\盘根目录,建立 C:\COMP51 子目录(文件夹)
2.将第三方的 51 编译器复制到 C:\COMP51 子目录(文件夹)下.
3.在 [主菜单 | 仿真器 | 仿真器设置 | 语言] 对话框的
[编译器路径] 指定为 C:\COMP51 (参见 仿真器设置)
☺ 如果用户将第三方编译器安装在硬盘的其它位置,请在[编译器路径]指明其位置.
例如:“C:\KEIL\C51\”

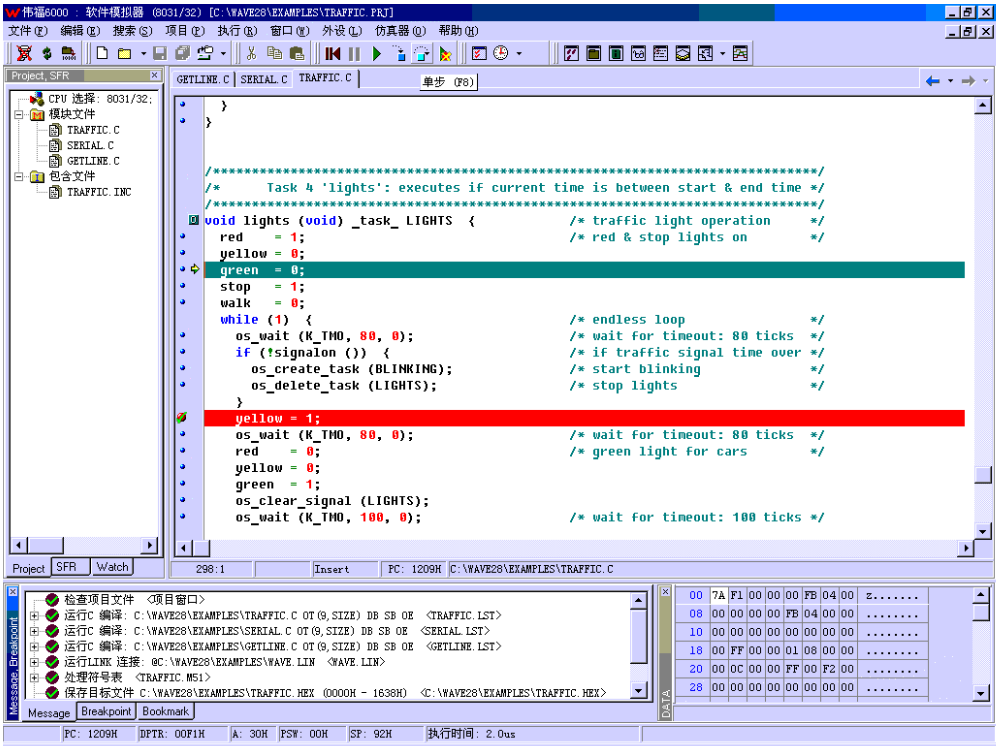
安装 96 系列 CPU 的编译器

1.进入 C:\盘根目录,建立 C:\COMP96 子目录(文件夹)
2.将第三方的 96 编译器复制到 C:\COMP96 子目录(文件夹)下.
3.在 [主菜单 | 仿真器 | 仿真器设置 | 语言] 对话框的
[编译器路径] 指定为 C:\COMP96 (参见 仿真器设置)
☺ 如果用户将第三方编译器安装在硬盘的其它位置,请在[编译器路径]指明其位置.

安装 PIC 系列 CPU 的编译器

1 进入 C:\盘根目录,建立 C:\COMPPIC 子目录(文件夹)
2 将第三方的 96 编译器复制到 C:\COMPPIC 子目录(文件夹)下.
3 在 [主菜单 | 仿真器 | 仿真器设置 | 语言] 对话框的
[编译器路径] 指定为 C:\COMPPIC (参见 仿真器设置)
☺ 如果用户将第三方编译器安装在硬盘的其它位置,请在[编译器路径]指明其位置.

4 开发环境



文件(F)

文件 | 打开文件

打开用户程序，进行编辑。如果文件已经在项目中,可以在项目窗口中双击相应文件名 打开文件.

文件 | 保存文件

保存用户程序。用户在修改程序后,如果进行编译,则在编译前,系统会自动将修改过的文件存盘.

文件 | 新建文件

建立一个新的用户程序， 在存盘的时候,系统会要求用户输入文件名.

文件 | 另存为

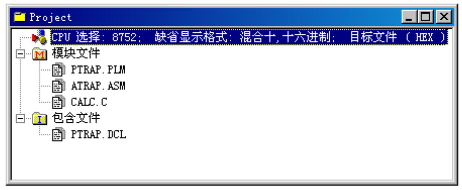
将用户程序存成另外一个文件， 原来的文件内容不会改变

文件 | 重新打开

在重新打开的下拉菜单中有最近打开过的文件及项目，选择相应的文件名或项目名就可以重新打开文件或项目。

文件 | 打开项目

打开一个用户项目，在项目中，用户可以设置仿真类型。加入用户程序，进行编译，调试。系统中只允许打开一个项目，打开一个项目或新建一个项目时，前一项目将自动关闭。



（图：项目窗口）

伟福开发环境的项目文件包括仿真器设置,模块文件,包含文件.

仿真器设置包括仿真器类型,仿真头(POD)类型,CPU 类型,显示格式和产生的目标文件类型

可以用以下几种方法设置仿真器.

- o 在项目窗口中双击第一行,将打开仿真器设置窗口,对仿真器进行设置.
- o 按鼠标右键,在弹出菜单中选择[仿真器设置].
- o 主菜单 仿真器|仿真器设置.

加入模块文件

- o 按鼠标右键,在弹出菜单中选择[加入模块文件]
- o 主菜单 项目|加入模块文件

加入包含文件

- o 按鼠标右键,在弹出菜单中选择[加入包含文件]
- o 主菜单 项目|加入包含文件

用户可以将以前单文件方式仿真转为 WINDOWS 下的项目方式进行仿真

1. 主菜单 文件|新建项目,在新建项目时,前一个项目自动关闭.
2. 加入模块文件时,选择要调试的程序文件名,将文件加入项目.
3. 将项目存盘.
4. 编译,运行,调试项目.

文件 | 保存项目

将用户项目存盘。用户在编译项目时,自动存盘。　　注意:

当用项目仿真时,系统要求项目文件,模块文件,包含文件在同一个目录(文件夹)下.

文件 | 新建项目

当用户开始新的任务时，应新建一个项目，在项目中，设置所用仿真器类型，POD 类型，加入用户程序（模块）。

文件 | 关闭项目

关闭当前项目，如果用户不想用项目方式调试单个程序，就要先关闭当前项目。

文件 ｜ 项目另存为

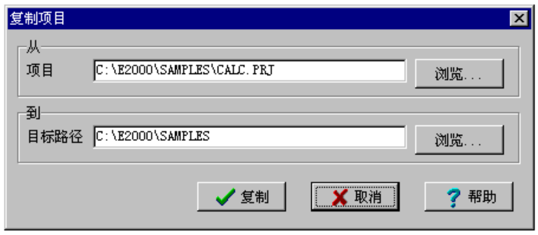
将项目换名存盘,此方法只是将项目用另一个名字,而不会将项目中的模块和包含文件换成另一个名字存盘.如果想将整个项目及模块存到另一个地方,请用复制项目方法.

文件 ｜ 复制项目

复制项目，用户可以将项目中的所有模块（用户程序）备份到另一个地方。在多模块项目中,用复制项目功能,可以避免用户因为少复制某些模块,而造成项目编译不能通过.方便用户对程序进行管理.

复制项目对话框.

[**从项目**]栏中为当前被复制项目,包括项目中各模块,包含文件,如果不是复制当前项目,可以通过[**浏览**]找到所要复制的项目,[**到目标路径**]中为项目复制到何处,可以通过其后的[**浏览**]指定将项目复制到其它地方.



文件 ｜ 调入目标文件

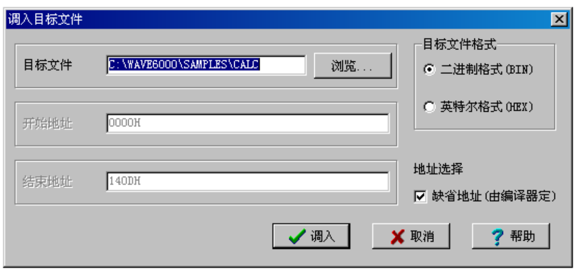
装入用户已编译好目标文件。系统支持两种目标文件格式：BIN，HEX 格式

调入已经编译好的目标文件

目标文件格式有二种:

二进制(BIN):由编译器生成的二进制文件,也就是程序的机器码

英特尔格式(HEX):由英特尔定义的一种格式,用 ASCII 码来存储编译器生成的二进制代码,这种格式包括地址,数据和校验



（图：调入目标文件对话框）

地址选择一般为缺省地址(由编译器定).如果想在当前项目已编译好的二进制代码中插入一段其它代码,可以去掉"缺省地址"前的选择.然后填入开始插入的地址和结束地址。用调入目标文件的方法,可以调试已有的二进制代码程序.而不需要源程序.

直接调入目标文件进行仿真的方法是:

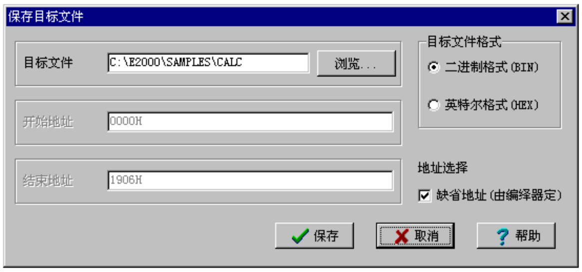
1.关闭项目。

2. 在新建的项目中,设置仿真器类型,仿真头类型,CPU 类型.
3. 调入目标文件.(不要用加入模块方式,而是直接调入文件)
4. 打开 CPU 窗口,在 CPU 窗口中就可以看见目标文件反汇编生成的程序.
5. 程序停在与 CPU 相关的地址上(51 系列停在 0000H 处,96 系列停在 2080H)
6. 这样就可以单步或全速调试程序了.

文件 | 保存目标文件

将用户编译生成的目标文件存盘。

对于按项目方式仿真的用户,系统将程序编译正确后,会根据用户在 仿真器|仿真器设置下[目标文件] 中设置的格式,将生成的二进制代码存盘. 如果用户是用调入目标文件方式进行仿真,并且对目标码进行了修改.就可以用 文件|保存目标文件 方式存盘.



目标文件可以存成两种格式:

[**二进制格式(BIN)**]:由编译器生成的二进制文件,也就是程序的机器码

[**英特尔格式(HEX)**]:由英特尔定义的一种格式,用 ASCII 码来存储编译器生成的二进制代码,这种格式包括地址,数据和校验

[**地址选择**]一般为[缺省地址(由编译器定)].如果想要存盘的目标文件是由“调入目标文件”方式装入,而不是由系统编译产生的代码,并已经修改,最好指定它的开始地址和结束地址,因为代码不是编译系统产生的.系统不知道文件有多长,无法指定开始和结束地址。自己指定地址的方法是:去掉[缺省地址]前的选择勾.然后填入开始插入的地址和结束地址。

文件 | 反汇编

将可执行的代码反汇编成汇编语言程序。(详见伟福反汇编功能的使用方法)

文件 | 打印

打印用户程序。

文件 | 退出

退出系统,如果在退出以前有修改过的文件没有存盘,系统将会提示是否把文件存盘.

编辑(E)

编辑 | 撤消键入
取消上一次操作
编辑 | 重复键入
恢复被取消的操作
编辑 | 剪切
删除选定的正文，删除的内容被送到剪贴板上
编辑 | 复制
将选定的内容，复制到剪贴板上
编辑 | 粘帖
将剪贴板的内容插入光标位置
编辑 | 全选
选定当前窗口所有内容。

搜索(S)

搜索 | 查找
在当前窗口中查找符号，字串。可以指定区分大小写方式，全字匹配方式，可以向上／向下查找。
搜索 | 在文件中查找
可以在指定的一批文件中查找某个关键字。
搜索 | 替换
在当前窗口查找相应文字，并替换成指定的文字，可以指定区分大小写方式和全字匹配方式查找，可以在指定处替换，也可以全部替换。
搜索 | 查找下一个
查找文字符号下一次出现的地方
搜索 | 项目中查找
在项目所有模块（文件）中查找符号，字串。在项目所包含的文比较多时，用此方法可以很方便地查到字串在什么地方出现。
搜索 | 转到指定行
将光标转到程序的某一行。
搜索 | 转到指定地址/标号
将光标转到指定地址或标号所在的位置。
搜索 | 转到当前 PC 所在行
将光标转到 PC 所在的程序位置。

项目(P)

项目 | 编译

编译当前窗口的程序。如有错误，系统将会指出错误所在的位置。

项目 | 全部编译

全部编译项目中所有的模块（程序文件），包含文件。如有错误
系统会指出错误所在位置。

项目 | 装入 OMF 文件

建好项目后，无须编译，直接装入在其它环境中编译好的调试信息，
在伟福环境中调试。

项目 | 加入模块文件

在当前项目中添加一个模块。

项目 | 加入包含文件

在当前项目中添加一个包含文件

执行(R)

执行 | 全速执行

运行程序

执行 | 跟踪

跟踪程序执行的每步，观察程序运行状态。

执行 | 单步

单步执行程序，与跟踪不同的是，跟踪可以跟踪到函数或过程的内
部，而单步执行则不跟踪到程序内部。

执行 | 执行到光标处

程序从当前 PC 位置，全速执行到光标所在的行。如果光标所在行没
有可执行代码。则提示"这行没有代码"

执行 | 暂停

暂停正在全速执行的程序。

执行 | 复位

终止调试过程，程序将被复位。如果程序正在全速执行，则应先停止。

执行 | 设置 PC

将程序指针 PC，设置到光标所在行。程序将从光标所在行开始执行。

执行 | 自动单步跟踪/单步

模仿用户连续按 F7 或 F8 单步执行程序。

执行 ｜ 编辑观察项

观察变量或表达式的值，可以将需要检查和修改的值或表达式放到观察窗口里以便检查和修改。 （图：观察项对话框）



[表达式]： 用于输入用户所要求值的表达式.

[重复次数]:如果表达式为某一存储变量,重复次数表示以此变量开始的连续 N 个地址的值.

[显示格式]:指定用何种方式显示表达式的值.

[存贮区域]:指明变量所在的区域.

[显示类型]:指定表达式为何种类型的变量.

[缺省方式显示]: 按照高级语言定义的方式显示.

[存贮器内容]: 以内存方式显示观察内容,也就是按地址顺序显示变量值,与变量类型无关

[求值]: 对表达式求值,并按显示格式显示在窗口内.

[加入观察]:将表达式加入观察窗口中,以便随时察看.

[编辑观察]:当修改过窗口内容后,按此键后,替代观察窗口中的原观察项,如果选择[加入观察],则会在观察窗口中另加一个变量的观察项,以两种格式观察一个变量.

[取消]: 关闭编辑观察项窗口

执行 ｜ 设置/取消断点

将光标所在行设为断点，如果该行原来已为断点，则取消该断点。所有断点通过断点窗口进行管理。

四种方法可以在光标处设置断点

1. 将光标移到编辑窗口内,行左边的空白处,光标变成“手指圆”箭头,单击鼠标左键, 可以设置/取消断点.
2. 使用 Ctrl-F8 快捷键,可以在光标所在行设置/取消断点
3. 右击鼠标, 弹出菜单,选择 设置/取消断点,
4. 主菜单 执行/设置取消断点,也可以用 Alt-R / B 菜单快捷设置取消断点

执行 | 清除全部断点

清除程序中所有的断点。让程序全速执行

窗口(W)

窗口 | 刷新

刷新打开的所有窗口，及窗口里的数据。

窗口 | 项目窗口

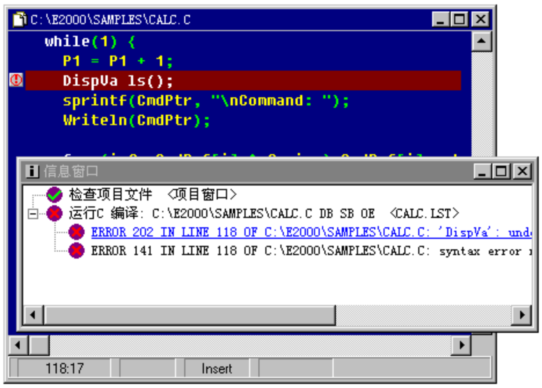
打开项目窗口，以便在项目中加入模块或包含文件。(图：项目窗口)

窗口 | 信息窗口

显示系统编译输出的信息。如果程序有错，会以图标形式指出，

表示错误，　表示警告，　表示通过

在编译信息行会有相关的生成文件，双击鼠标左键，或击右键在弹出菜单中选择"打开"功能，可以打开相关文件。(如果有编译错误，双击左键，可以在源程序中指出错误所在行，有时前一行或后一行程序有错，会造成当前行编译不通过。而将错误定位在本行，所以如果发现了错误，但在本行没有发现错误，可以查查本行上下的程序)。



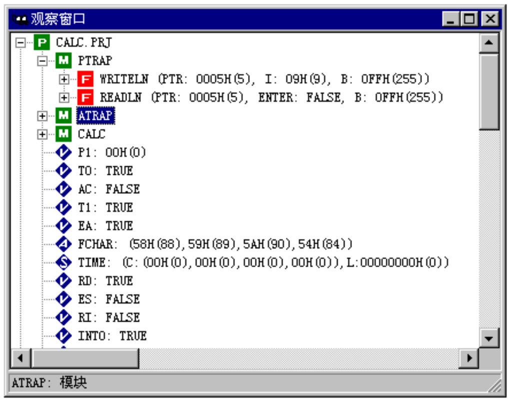
例：编译过程发现有错。在信息窗口中看到在 CALC.C 文件第 118 行有 202 号错误，

文字显示错误类型是，" 'DispVa' undefined identifier" 即：未定义 DispVa 标识符。 双击此信息行，系统将打开 CALC.C 文件，并且在源文件的 118 行，指出有错，可以看到，DispVa 和 ls()中间有空格。 原来应为 DispVals()。

窗口 | 观察窗口

项目编译正确后，可以在观察窗口中看到当前项目中的所有模块，及各模块中的所在过程和函数，及各个过程函数中的各个变量，结构。如果能充分利用观察窗口的强大功能，可以加快你开发速度。

观察窗口也可以用观察数据时效分析,程序时效分析,代码覆盖以及影子存储器等分析功能的结果.(参见各分析功能使用)

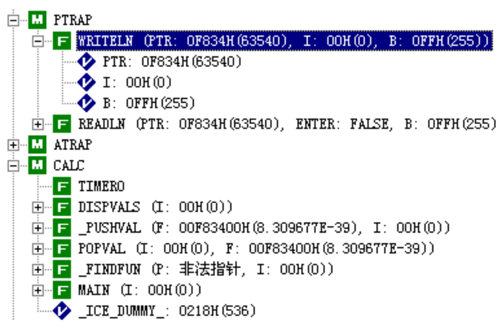


（图：观察窗口）

P 表示当前项目,双击可以展开,观察到项目中的模块和项目所使用的变量

M 表示项目中所包括的模块.双击可以展开,观察到项目中包含的过程函数.

例:一个打开的项目,可以看项目中包括 PTRAP, ATRAP, CALC 三个模块,其中 PTRAP 展开,PTRAP 包括 WRITELN 和 READLN 两个函数,可以看到展开的 WRITELN 函数使用个三个简单变量:PTR,I,B. 在展开的CALC模块中包含了六个函数.



F 表示模块中的函数,双击可以观察到模块中所用到的变量.

表示模块或函数中使用的简单变量.

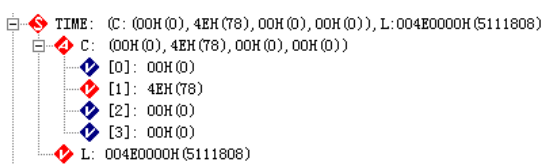
表示模块或函数中使用的数组,双击可以展开数组,观察数组中各值的变化.

表示模块或函数中使用的结构,双击可以展开结构,观察结构内部变量值.

P 表示模块或函数中使用的指针

表示模块或函数中使用的标号

例: 一个展开的结构.结构变量名为 TIME,它包括一个数组变量和一个长整形变量,通过展开的数组,可以看到数组有四个元素,其中第二个元素在上次执行过程中发



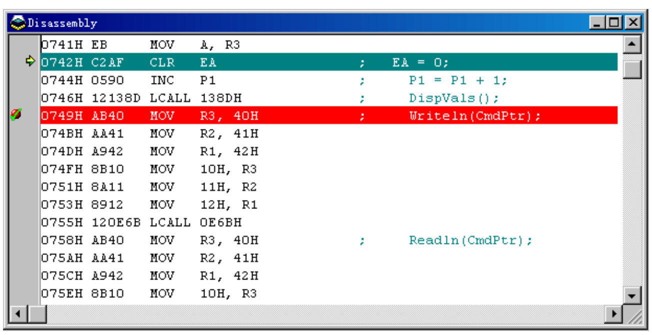
生了变化,长整形变量 L 也发生变化.
标记颜色为红,表示在上次执行过程中变量值发生变化.
标记颜色为橄榄色,表示在上次执行过程中,变量被访问过.

可以在弹出菜单中选择
[加入观察], 把当前行的变量放在窗口的最后,而不用展开复杂的项目,模块来观察某个变量.
[编辑观察项],可以打开"编辑观察窗口",对当前变量进行修改,求值
[删除观察项],删除观察窗口后变量.不再对此变量进行观察.
[察看源程序],若窗口中当前行是模块文件,用此命令可以打开相应的模块文件.
[展开],[收缩],用于展开/收缩当前行的函数,结构,数组.便于观察.
[修改], 对当前行的变量进行修改,在程序运行时,给出所要求的值.以观察程序在此值时,运行的结果.
[窗口总在最上面],若此项被选中,则窗口会一直显示在前面,覆盖其它窗口. 建议选中此项,使窗口在最上面时,不要使窗口最大化,以免完全覆盖其它窗口.

在观察窗口最下面的状态行可以看到观察项的更详细信息,当你在点击一个项目时,可以看到该观察项的类型(模块,函数/过程),所在区域(CODE,DATA,BIT,XDATA 等到),地址,数据类型(unsigned char, unsigned int, real 等)

窗口 | CPU 窗口

通过 CPU 窗口,可以打开反汇编窗口,SFR 窗口和 REG 窗口。在反汇编窗口中可观察编译正确的机器码及反汇编程序,可以让你更清楚地了解程序执行过程。SFR 窗口中可以观察到单片机使用的 SFR(特殊功能寄存器)值和位变量的值。REG 窗口为 R0..R7、A、DPTR 等常用寄存器的值。



反汇编窗口内为程序地址,机器码,反汇编码。在机器码窗口内也支持点屏功能,在反汇编码处,点击寄存器,可以看到寄存器的值.
反汇编窗口的弹出菜单
执行到光标处 : 使程序从当前 PC 值,全速执行到光标所在行,用这种方法可以在调试程序时,跳过一些不必要的指令.将程序停到所要求的位置上.

转到指定地址/标号: 将光标跳到某个地址或标号所在位置,以便察看相应的程序,或使用 “执行到光标处”功能,也可以设置断点, ,将程序全速执行到相应位置.

转到当前 PC 所在行: 将光标跳到 PC 所在行,由于在检查程序时,可能会将 PC 所在行移出当前窗口,用这种方法可找回 PC 所在行.

取消/设置断点: 在光标所在行,设置断点,使程序全速执行到此处.若此行已是断点,再次点击将取消该断点.

寄存器窗口的弹出菜单

加入观察: 将当前寄存器放入观察窗口,以方便随时察看.

修改: 修改当前寄存器值.在程序执行时,可以用这种方法,把寄存器值改为你所指定的值,从而观察程序在此值时运行的结果.

窗口 | 数据窗口

数据窗口根据选择的 CPU 类型不同, 名称有所不同。

51 系列有以下四种数据窗口

DATA 内部数据窗口　　CODE 程序数据窗口

XDATA 外部数据窗口　　PDATA 外部数据窗口（页方式）　　BIT 窗口

196 系列有以下三种数据窗口

REGISTER 寄存器窗口

CODE 程序数据窗口

DATA 数据窗口

PIC 系列有以下三种数据窗口

EEPROM 电擦写数据窗口

DATA 数据窗口

CODE 程序数据窗口

以 51 系列为例说明数据窗口的操作方法,其它 CPU 类型的数据窗口基本相同

内部数据窗口

在内部数据窗口中可以看到 CPU 内部的数据值, 红色的为上一步执行过程中, 改变过的值, 窗口状态栏中为选中数据的地址, 可以在选中的数据上直接修改数据的十六进制值, 也可以用弹出菜单的修改功能, 修改选中的数据值。



弹出菜单：

修改：修改选中数据的值，可以输入十进制，十六进制，二进制的值，与直接修改不同的是，用这种方法可以输入多种格式数据，而直接修改只能输入十六进制数据。46（十进制），2EH（十六进制），00101110B（二进制）都是有效的数据格式，

转到指定地址/标号：将数据地址直接转到指定的地址和标号所在的位置。

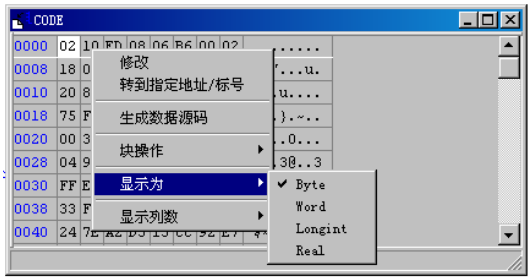
生成数据源码：将窗口中某段数据转换成源程序方式的数据，可以贴到你的源程序中。

块操作：对窗口中的数据块进行填充、移动、写文件、读入等操作。

显示为：选择不同的数据类型显示数据内容，可以是字节方式（BYTE），也可以是字方式（WORD，两字节），可以是长整型（LONGINT，四字节），也可以是实数型（REAL，四字节）。这里是选择整个窗口的显示方式，如果想指定个别数据的显示方式，可以用主菜单[执行|编辑观察项]功能，选择所要选择的显示类型。（参见编辑观察项窗口）

显示列数：将窗口中数据以 4 列、8 列、16 列方式显示。适应不同需要。

程序数据窗口显示的是编译后程序码，状态栏显示的是选中数据的地址，可以对在选中数据上直接修改程序数据的十六进制值，也可以对程序数据进行‘块填充’，‘块移动’操作，也可以读入一段二进制代码插入程序数据中，也可以将程序数据中的某段代码写文件中.



程序数据窗口

弹出菜单

修改：修改选中数据的值，可以输入十进制，十六进制，二进制的值，与直接修改不同的是，用这种方法可以输入多种格式数据，而直接修改只能输入十六进制数据。46(十进制)，2EH(十六进制)，00101110B(二进制)都是有效的数据格式.

生成数据源码：将窗口中某段数据转换成源程序方式的数据，可以贴到你的源程序中。

块操作: 对程序数据以块的方式进行操作.在窗口中按往左键拖动,可以选择块。

块填充:将选中的块内数据值,填充为指定值.

块移动:将选中的块移动到指定地址.

读文件:读入二进制代码文件,插入的指定的地址内.(参见‘调入目标文件’功能)

写文件:将程序数据指定地址的一段代码写入文件.(参见‘保存目标文件’功能)

显示为：选择不同的数据类型显示数据内容，可以是字节方式（BYTE），也可以是字方式（WORD，两字节），可以是长整型（LONGINT，四字节），也可以是实数型（REAL，四字节）。这里是选择整个窗口的显示方式.

窗口 | 断点窗口

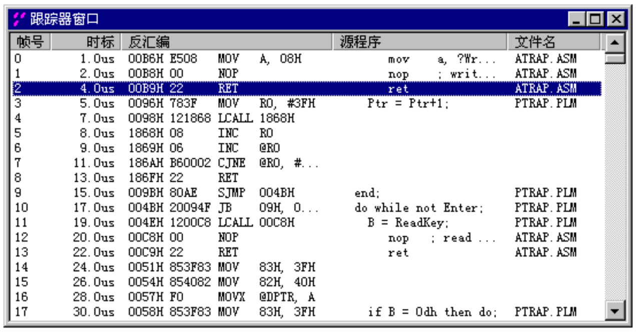
通过断点窗口可以管理项目内的断点。可以在断点窗口中直观地看到断点的行号，内容，可以通过断点迅速定位程序所在的位置。

窗口 | 书签窗口

通过书签窗口可以管理项目内的书签，在项目中迅速定位程序位置。

窗口 | 跟踪窗口

显示跟踪器捕捉到的程序执行的轨迹，其中可以看到帧号，时标，反汇编程序，对应的源程序和程序所在的文件名。



窗口 | 逻辑分析窗口

在这窗口中观察到逻辑分析仪所采集到的波形，可以设置不同的采样方式，以满足各种情况下的需要。逻辑分析仪是数字设计中不可缺少的设备，通过它，可以清楚地看到程序执行时，各端口输出的波形，迅速地帮助你找出硬件和软件中设计错误。

窗口 | 工具条

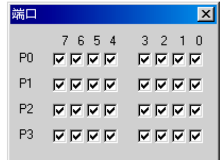
通过工具条，可以打开/关闭菜单上的各功能的快捷按钮。

窗口 | 排列窗口

对打开的程序窗口进行管理。 可叠排、坚排、横排、最小化源程序窗口。

外设(L)

外设｜端口



设置或观察当前端口的状态。

外设｜定时器/计数器 0

定义或观察定时器/计数器 0，通过定义定时器/计数器的工作方式，自动生成相应的汇编/C 语言。可以“复制/粘贴”到你的程序中。



T0/T1 定时器/计数器



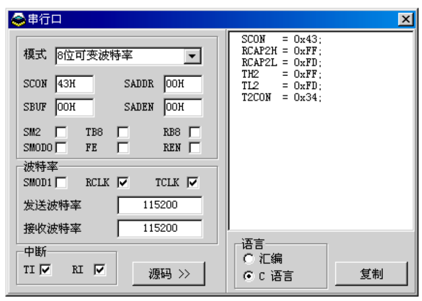
T2 定时器/计数器

外设｜定时器/计数器 1

定义或观察定时器/计数器 1，通过定义定时器/计数器的工作方式，自动生成相应的汇编/C 语言。可以“复制/粘贴”到你的程序中。

外设｜定时器/计数器 2

定义或观察定时器/计数器 2，通过定义定时器/计数器的工作方式，自动生成相应的汇编/C 语言。可以“复制/粘贴”到你的程序中。



外设｜串行口

定义或观察串行口的工作方式，可以观察串行口的工作方式是否正确，也可以定义串口的工作方式，自动生成串口初始化程序。(串口的波特率的时钟为仿真器设置中“使用伟福软件模拟器”的晶体频率，见“仿真头设置”)

外设｜中断

管理或观察中断源，也可以辅助生成中断初始化程序。



仿真器(O)

仿真器 | 仿真器设置

语言设置

设置项目编译语言的路径，命令行选项。



当 CPU 为 51 芯片时,语言设置对话框



当 CPU 为 196 时,语言设置对话框

[编译器路径]:指明本系统汇编器,编译器所在位置,系统缺省 51 系列编译器在 C:\COMP51\文件夹下,缺省 96 系列编译器在 C:\COMP96\文件夹下.本系统使用的编译器为第三方软件,你应从其它途径获得.

[ASM 命令行]:若使用英特尔汇编器,则需要加上所需的命令行参数。若使用伟福汇编器,则需要选择是否使用伟福预定义的符号.在伟福汇编器中已经把 51/96 使用的一些常用符号,寄存器名定义为相应的值.如果你使用伟福汇编器,就可以直接使用这些符号.如果你自己已经定义了这些符号,又想使用伟福汇编器,就将“使用伟福预定义符号”前面的选择去掉.

[C 命令行]: 项目中若有 C 语言程序,系统进行编译时,使用此行参数对 C 程序进行编译.

[PL/M 命令行]:项目中若有 PL/M 语言程序,系统编译时,就使用此行参数对程序进行编译.

[LINK 命令行]:系统对目标文件链接时,使用此参数链接.

注: 除非你对命令行参数非常了解,并且确实需要修改这些参数,一般情况下,不需要修改系统给出的缺省参数.以免系统不能正常编译.

[编译器选择]: 选择使用伟福汇编器,还是英特尔汇编器,系统对 C 语言程序和 PL/M 语言编译是采用第三方编译器. 一般情况下,如果用户项目中都是汇编语言程序,没有 C 语言和 PL/M 语言,选择伟福汇编器. 如果用户项目中含有 C 语言,PL/M 语言,或者汇编语言是用英特尔格式编写的,就选择英特尔汇编器.

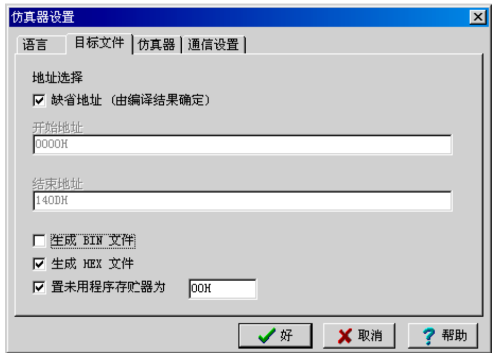
[缺省显示格式]:

指定观察变量显示的方式,一般为混合十/十六进制.



当 CPU 为 MICROCHIP 芯片时,可以选择是用伟福汇编器还是选择 MICROCHIP 公司的汇编器及 HT-PICC 的 C 语言。若是 MICROCHIP 汇编器，ASM 命令行的缺省如右图。右是 HT-PICC 语言，C 命令行的缺省如右图。

目标文件设置



设置生成的目标文件的地址，及生成目标文件的格式。

一般情况下,地址选择为缺省方式.即由编译确定. 如果你想重新定位你的程序就要指定地址,方法是:去掉[缺省地址]前面的选择. 在开始地址,结束地址处填入相应的地址. 编译可以生成 BIN(二进制)格式和 HEX(英特尔)格式的目标文件,可以根据你的需要,选择相应的格式。

仿真器设置



选择所使用的仿真器类型，POD（仿真头）类型，以及所仿真 CPU 的类型。如果使用硬件仿真，请去掉“使用伟福软件仿真”前的选择，在仿真头设置中可以设置该仿真头的特殊功能。包括仿真空间，看门狗，加密位等等

[**选择仿真器**]:框内为本系统所支持的仿真器类型.选择正确仿真器.

[**选择仿真头**]:　框内为相应仿真器能支持的仿真头类型,选择所使用的仿真头.

[**选择 CPU**]:　框内为选择的仿真器和仿真头能等进行仿真支持的 CPU.

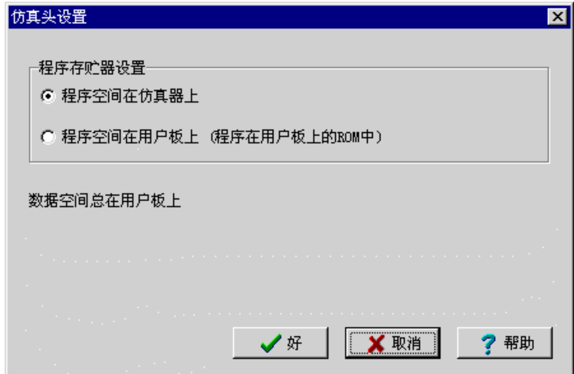
[**使用伟福软件模拟器**]:使用伟福软件模拟器,可以在完全脱离硬件仿真器情况下,对软件进行模拟执行. 如果使用硬件仿真器,请不要选择使用伟福软件.

[**晶体频率**]: 在使用伟福软件模拟功能时，用来计算在软件模拟环境下程序执行时间。
在外设中串行口的波特率也是依据此频率计算出的。

[**仿真头设置**]: 可以设置该仿真头的特殊功能。包括仿真空间，看门狗，加密位等等.
仿真头(POD)类型不同,设置内容有所不同. (见仿真头设置)

POD51,POD80552 仿真头设置

如果用户已经将调试好的程序,烧到EPROM 中,插到用户板上,想观察程序在用户板上工作是否正常.请将设置选到[程序空间在用户板上]。通常情况下,用户在调试程序时,请将设置选到[程序空间在仿真器上]



POD 80196 仿真头设置



程序地址前面的勾表示相应地址段的程序在仿真器内,正常调试时,应该所有地址都设在仿真器内.但 I/O 地址空间和 RAM 空间必须设置在用户板上。如果用户程序已经调试正确,并且已经烧到 EPROM 中插到用户板上,根据程序地址和大小将相应地址前的选择勾去掉,用户就可以执行用户板上 EPROM 中程序了。当用户使用 INST 控制功能时,如果相应地址被选中,这段地址在仿真内部,INST 不参加地址译码,如果地址没有被选中,程序在仿真器内部,数据在用户板上.

[**掉电使能**]:　　为 0 时可防止 CPU 意外地进入掉电(睡眠)方式

[**总线宽度控制**]:　　选择总线宽度控制方式

[**地址有效选择**]:　　选择地址有效方式

[**(WDTE)看门狗使能**]: 决定程序工作时看门狗是否工作。

[**写控制**]:　　选择写控制方式

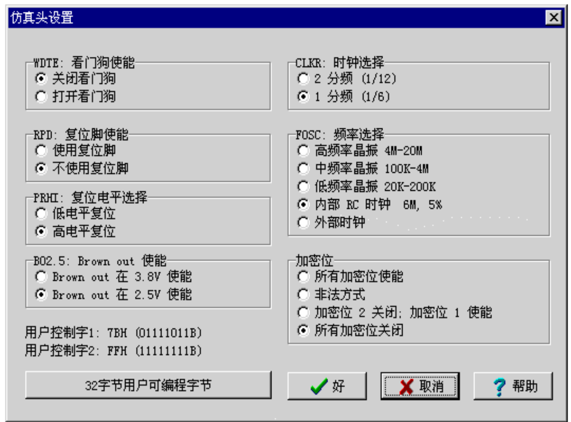
[**等待控制**]:　　选择芯片等待外部存贮器就绪的时间

[**加密位使能**]:　　选择 87C196 的加密方式.

更详细的说明请参见 INTEL 公司的 8XC196 芯片资料.

如果按照以上方式,定义好后,系统已经将控制字写入 2018H 及 201AH(MC/MD)单元,即使用户在程序中自己定义控制字,系统并不采用,而是用此对话框设置为准,所以用户在仿真时和生成目标代码时,请用此对话框设置 196 系列的控制字.

POD LPC76X 仿真头设置



[**(WDTE)看门狗使能**]: 决定程序工作时看门狗是否工作,在调试程序时,请关闭看门狗,否则在调试程序时,看门狗会工作，若不及时清除看门狗定时器，程序会被复位。可以用这种方法调试 LPC76X 的看门狗，方法是在程序开始处设置断点，程序运行，看门狗造成程序停止后，观察看门狗寄存器。

[**(RPD)复位脚使能**]:　LPC76X 系列可以将复位脚的复位功能关闭,而把此管脚用作 IO 口,根据你的硬件设计,做适当选择.

[**(PRHI)复位电平选择**]:选择 CPU 复位时,I/O 端口电平是处于高电平状态还是低电平状态。适应用户不同的电路设计需要.

[**(BO2.5)Brown out 使能**]: 选择当电源电压低于 3.8V 时复位,还是低于 2.5V 时复位.

[**(CLKR)时钟选择**]: 选择 LPC76X 工作速度为普通 51 系列一倍还是两倍. 在 LPC76X 内部,1 个机器只要 6 个振荡周期,而不是象普通 51 系列 CPU 要 12 个振荡周期,所以在相同时钟频率下,LPC76X 比普通 51 快一倍. 也可以通过选择此项,使其工作速度与普通 51 相同。

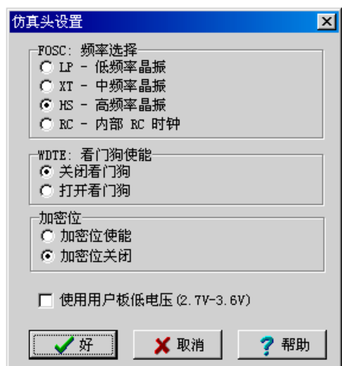
[**(FOSC)频率选择**]: 根据用户板上的晶振频率,选择 CPU 工作频率的范围.

[**加密位**]:程序调试正确后,将程序烧到芯片中,选择程序加密方式.

[**32 字节用户可编程字节**]: 用于存储用户程序以外的信息,例如版本号,系列号等

更详细的说明请参见 PHILIPS 公司的 LPC76X 芯片资料.

POD PIC 系列仿真头设置:



[**(FOSC)频率选择**]: 根据用户目标板晶振频率不同,选择晶振的频率范围。对于 MICROCHIP 早期的仿真芯片，4MHz 晶振认为是中频率晶振。而对于现在的仿真芯片，4MHz 则认为是高频率晶振。

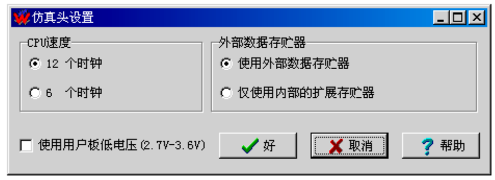
[**(WDTE)看门狗使能**]: 决定程序工作时看门狗是否工作,在调试程序时,请关闭看门狗,否在调试程序时,会工作不正常,出现莫明其妙的错误。

[**加密位**]:程序调试正确后,选择程序是否以加密方式烧到芯片中。

如果用户在仿真低电压时，请选中“使用用户板低电压（2.7V-3.6V)”。

更详细的说明请参见 MICROCHIP 公司的 PIC 芯片资料.

PODH8X5X/PODH591 系列仿真头设置:



PHILIPS 公司的很多芯片可以工作在 6 时钟/周期，[**CPU 速度**]用于选择 12 时钟/周期还是 6 时钟/周期。在这里有一点需要注意，当你选择 6 时钟还是 12 时钟时，同时也要用编程器对 CPU 的配置字进行编程，使其工作于相应的时钟下，例如，当你选择 6 时钟/周期时，应先对仿真头上的仿真 CPU 进行编程，擦除里面的程序，将芯片的工作时钟烧成 6 时钟/周期。PHILIPS 有一些芯片缺省工作方式为 12 时钟/周期，有一些芯片为 6 时钟/周期，要注意的是：有的芯片时钟选择位只能编程一次，不可恢复，在使用前请阅读芯片资料。有的芯片内部带有扩展 RAM，象 P89C51Rx2，P89C66x 等芯片，当使用芯片内部的扩展 RAM 时，请选择[**外部数据存贮器**]下的 “仅使用内部的扩展存贮器”，这样打开外部数据（XDATA）窗口观察外部数据时，就不会影响到 P0/P2 端口。当用户需要仿真低电压工作环境时，除了在硬件上要设置仿真头的跳线外（见仿真头介绍），还要在此选择“使用用户板低电压（2.7V-3.6V）”,另外仿真头上的仿真芯片要能工作于低电压状态。（**详见 PODH8X5X 使用说明**）

POD LPC93X 仿真头设置

PODLPC93X 仿真头的设置与 LPC76X 仿真头的设置相似。添加了一些扇区加密及 ISP 启动的设置。具体的设置可参考 LPC93X 的芯片资料。



通信设置



仿真器与计算机通信设置。包括通信端口选择，速率选择，字间距选择，以及串口的测试功能。如果选择了“使用伟福软件仿真”，则不需要设置通信端口。

[**端口选择**]: 选择仿真器与计算机连接的串口号. 如果计算机与仿真器连接不上,请检查通信 端口是否选择正确.

[**波特率选择**]: 选择仿真器与计算机连接的速度. 如果在高速率时通信不流畅,请降低通信速率.

[**字符间隔**]: 选择通信时,字符与字符之间的间隙,如果在小间隔时,通信不是很流畅,请调到较大的间隔.

[**使用伟福软件模拟器**]: 如果选择此项,可以在完全脱离硬件仿真器情况下,对软件进行模拟执行. 如果使用硬件仿真器调试程序,请去掉[使用伟福软件模拟器]前的选择勾.

[**测试串行口**]:用来检测仿真器是否正确连接到计算机的串行口上.

仿真器 | 跟踪器/逻辑分析仪设置



记时器 ：在程序下面的状态栏可以看到程序执行的时间. 注意:在用硬件单步执行程序时,记时器显示的时间可能略高于实际值,这是因为仿真器在采样时间时加入了监控时间.在全速执行多条指令时, 监控时间可以忽略不计.

逻辑笔 ：通过逻辑笔可以方便地检测到电路的高低电平,脉冲频率和数量.

跟踪器 ： 通过跟踪器,可以方便地看到程序实际执行的过程,在跟踪器窗口中可以观察到程序执行时间,执行过的机器码,反汇编程序,源程序,源程序所在文件。跟踪程序动态执行过程,找出程序中一些不可预见的错误.

影子存储器:在程序执行过程中,可以动态地看到存储器的变化,XDATA 窗口和观察窗口约每 2 秒刷新一次,这样就可以看到存储器当前值.为程序动态调试提供了更有效的手段.

程序时效分析:分析程序中,各过程,函数执行的时间,执行次数,了解程序执行效率,可以优化程序,进一步改善程序性能.

数据时效分析:分析程序中,各变量,数据被访问的次数,访问频率,从而改善程序的结构.开发出更有效,更稳定的程序.

逻辑分析仪: 通过逻辑分析仪,可以看硬件工作时,各点的状态,直观地用波形一表达,更易

检查出硬件,软件设计中的错误.

波形发生器: 可以定义你所想要的波形,输出到指定点,观察输出点是否正确,相当于一个可以定制的数字信号发生器. 为硬件调试提供了方便,快捷的手段。

仿真器 | 静态测试

对于只能工作于总线方式的仿真头（例如：POD-51，POD-80C196），可以用静态测试来静态地设置地址总线，数据总线，读写控制线，ALE、PSEN 等控制线状态（高或低），配合逻辑笔或电压表，可以很快地查出各种硬件连线及逻辑错误。



196 静态测试



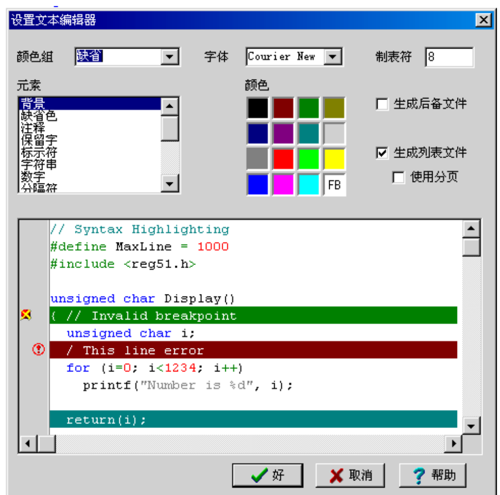
51 静态测试

仿真器 | 设置文本编辑器

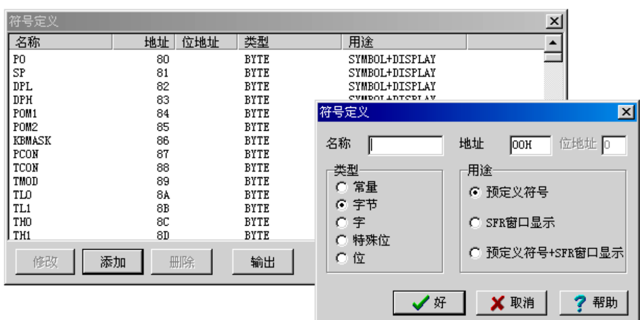
在设置文本编辑器的对话框中，你可以设置你自己喜爱的文本编辑环境。

仿真器 | 设置汇编预定义符号

在伟福开发环境中，用户可以自己定义或添加寄存器名称，按“添加”键来添加新的寄存器，在[**用途**]栏中，“预定义符号”是表示此符号用于伟福汇编器，“SFR 窗口显示”是表示在开发环境中的 SFR 窗口中，可以观察到此寄存器的值。



文本编辑器设置对话框



汇编预定义符号设置对话框及添加符号对话框

帮助(H)

帮助 | 关于

帮助 | CHINESE

选择中文或英文显示方式，适应不同操作系统的需要。

帮助 | 安装 MPASM

辅助用户安装 Microchip 的汇编器。将伟福 BIN 文件夹下的 MPASM 复制到指定的文件夹里。

快速入门

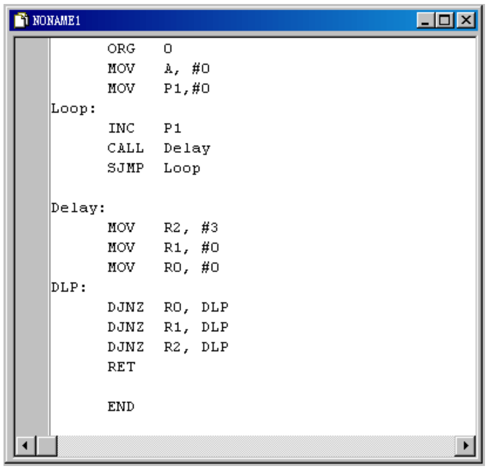
1．建立你的新程序

选择菜单[文件 | 新建文件]功能

出现一个文件名为 NONAME1 的源程序窗口，在此窗口中输入以下程序

```
      ORG   0
      MOV   A，#0
      MOV   P1，#0
Loop:
      INC   P1
      CALL Delay
      SJMP LOOP

Delay:
      MOV   R2，#3
      MOV   R1，#0
      MOV   R2，#0
DLP:
      DJNZ R0，DLP
      DJNZ R1，DLP
       DJNZ R2，DLP
       RET
       END
```



输出程序后的窗口如图，现在要做的是将此文件存盘。

2．保存你的程序

选择菜单[文件 | 保存文件]或[文件 | 另存为]功能

给出文件所要保存的位置，例如：C:\WAVE6000\SAMPLES 文件夹，再给出文件名 MY1.ASM。保存文件。文件保存后，程序窗口上文件名变成了：

C:\WAVE6000\SAMPLES\MY1.ASM

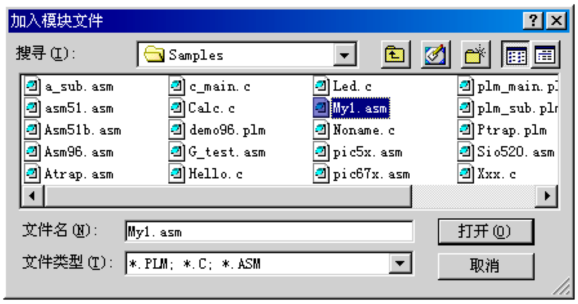


3. 建立新的项目

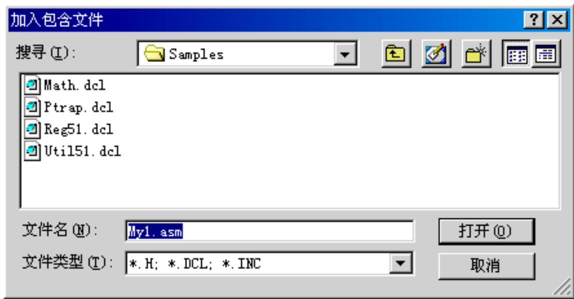
选择菜单[文件 | 新建项目]功能

新建项目会自动分三步走。

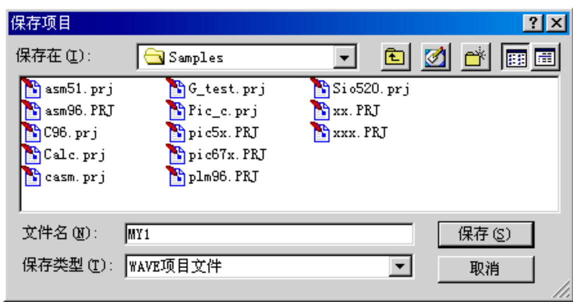
A） 加入模块文件。在加入模块文件的对话框中选择刚才保存的文件 MY1.ASM，按打开键。如果你是多模块项目，可以同时选择多个文件再打开。



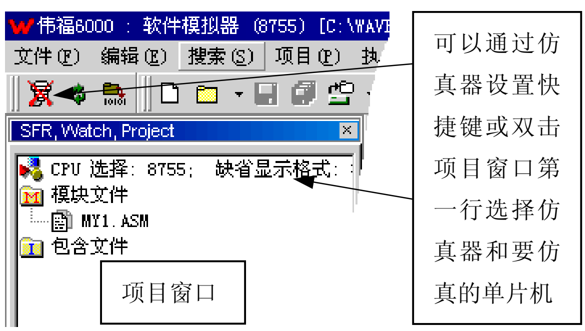
B） 加入包含文件。在加入包含文件对话框中，选择所要加入的包含文件（可多选）。如果没有包含文件，按取消键。



C） 保存项目。在保存项目对话框中输入项目名称。MY1 无须加后缀。软件会自动将后缀设成“.PRJ”。按保存键将项目存在与你的源程序相同的文件夹下。

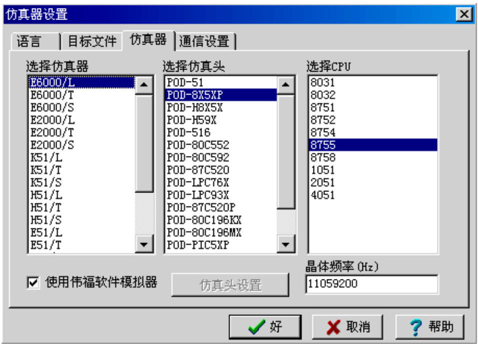


项目保存好后，如果项目是打开的，可以看到项目中的“模块文件”已有一个模块“MY1.ASM”，如果项目窗口没有打开，可以选择菜单[窗口 | 项目窗口]功能来打开。可以通过仿真器设置快捷键或双击项目窗口第一行选择仿真器和要仿真的单片机



4．设置项目

选择菜单[**设置 | 仿真器设置**]功能或按“仿真器设置”快捷图标或双击项目窗口的第一行来打开“仿真器设置”对话框

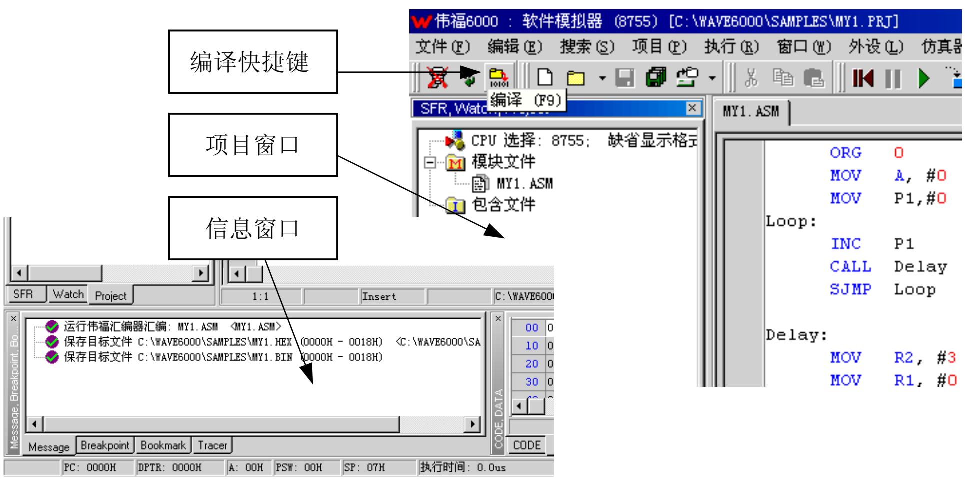




在“仿真器”栏中，选择仿真器类型和配置的仿真头以及所要仿真的单片机。在“语言”栏中，“编译器选择”根据本例的程序选择为“伟福汇编器”。如果你的程序是 C 语言或 INTEL 格式的汇编语言，可根据你安装的 Keil 编译器版本选择“Keil C (V4 或更低)”还是“Keil C (V5 或更高)”。按“好”键确定。当仿真器设置好后，可再次保存项目。

5．编译你的程序

选择菜单[**项目 | 编译**]功能或按编译快捷图标或 F9 键，编译你的项目。

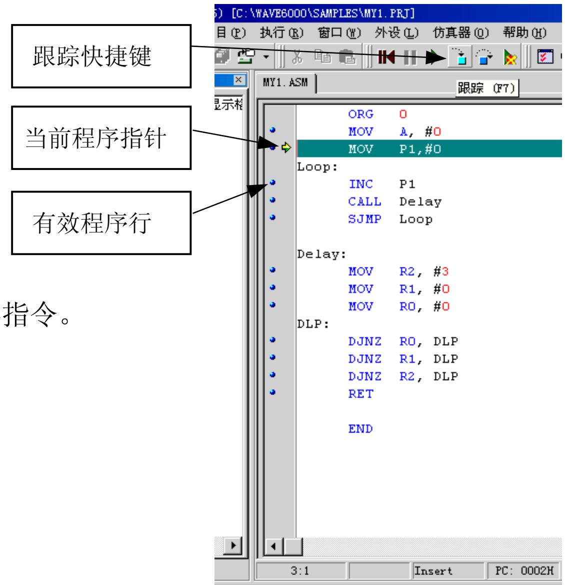


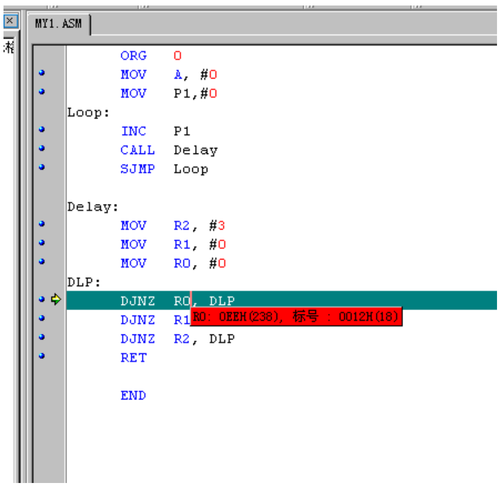
在编译过程中，如果有错可以在信息窗口中显示出来，双击错误信息，可以在源程序中定位所在行。纠正错误后，再次编译直到没有错误。在编译之前，软件会自动将项目和程序存盘。在编译没有错误后，就可调试程序了，首先我们来单步跟踪调试程序。

6. 单步调试程序

选择[执行 | 跟踪]功能或按跟踪快捷图标或按 F7 键进行单步跟踪调试程序

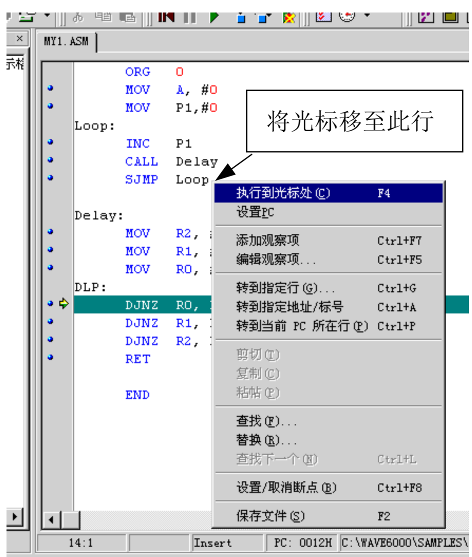
单步跟踪就一条指令一条指令地执行程序，若有子程序调用，也会跟踪到子程序中去。你可以观察程序每步执行的结果，“=>”所指的就是下次将要执行的程序指令。由于条件编译或高级语言优化的原因，不是所有的源程序都能产生机器指令。源程序窗口最左边的“o”代表此行为有效程序，此行产生了可以指行的机器指令。



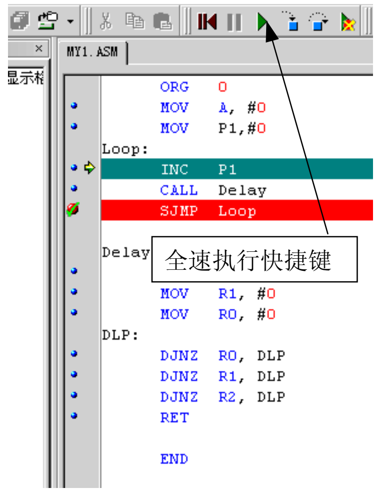


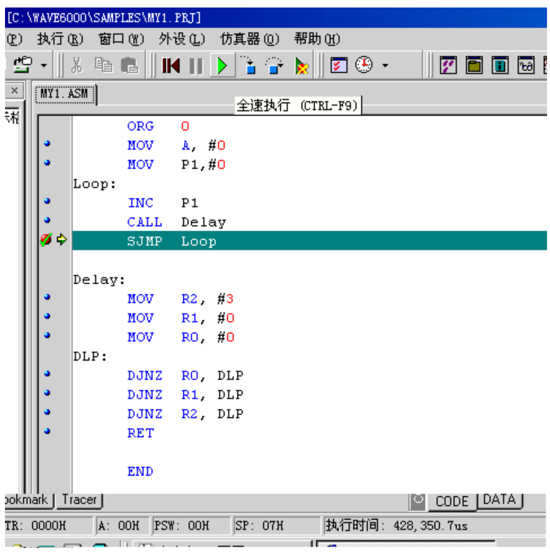
程序单步跟踪到“Delay”延时子程序中，在程序行的“R0”符号上单击就可以观察“R0”的值，观察一下“R0”的值，可以看到“R0”在逐渐减少。因为当前指令要执行 256 次才到下一步，整个延时子程序要单步执行 3x256x256 次才能完成，单步执行太慢了！

没关系，我们有“执行到光标处”的功能，将光标移到程序想要暂停的地方，本例中为延时子程序返回后的“SJMP Loop”行。选择菜单[执行 | 执行到光标处]功能或 F4 键或弹出菜单的“执行到光标处”功能。程序全速执行到光标所在行。如果想下次不想单步调试“Delay”延时子程序里的内容，可以按 F8 键单步执行就可以全速执行子程序调用，而不会一步一步地跟踪子程序。F8 F8F8F8F8F8F8......是不是太烦了？那就移动光标到暂停行再按 F4，如果程序太长，每次这样移来移去，是不是也太累？那就设置断点吧。



将光标移到源程序窗口的左边灰色区，光标变成“手指圈”，单击左键设置断点，也可以用弹出菜单的“设置/取消断点”功能或用 Ctrl+F8 组合键设置断点。如果断点有效图标为“红圆绿勾”，无效断点的图标为“红圆黄叉”。 断点设置好后，就可以用全速执行的功能，全速执行程序，当程序执行到断点时，会暂停下来，这时你可以观察程序中各变量的值，及各端口的状态，判断程序是否正确。



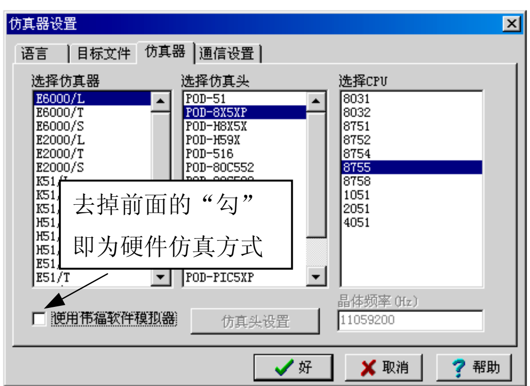


本例是将 P1 端口加一，然后延时，再重复，这样若 P1 就是一个二进制加法器，若 P1 口接发光二极管，就会闪亮。

不过到此为止，我们都是用软件模拟方式来调试程序。如果想要用仿真器硬件仿真。就要连接上仿真器。

7．连接硬件仿真

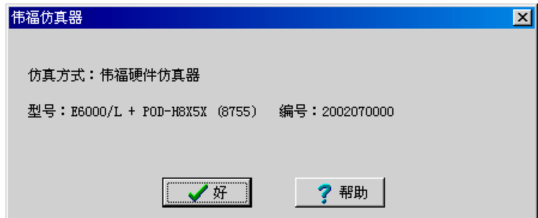
按照说明书，将仿真器通过串行电缆连接计算机上，将仿真头接到仿真器，检查接线是否有误，确信没有接错后，接上电源，打开仿真器的电源开关。



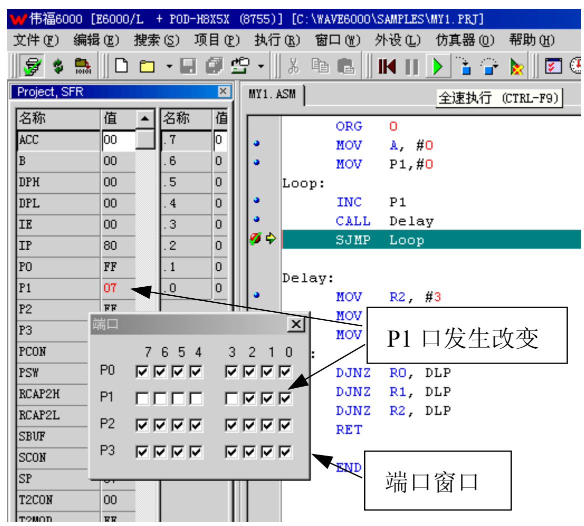


参见第 4 步，设置项目，在“仿真器”和“通信设置”栏的下方有“使用伟福软件模拟器”的选择项。将其前面框内的勾去掉。在通信设置中选择正确的串行口。按“好”确认。

如果仿真器和仿真头设置正确，并且硬件连接没有错误，就会出现如图的“硬件仿真”的对话框，并显示仿真器、仿真头的型号及仿真器的序列号。表明仿真器初始化正确。如果仿真器初始化过程中有错，软件就会再次出现仿真器设置对话框，这时你应检查仿真器、仿真器的选择是否有错，硬件接线是否有错，检查纠正错误后，再次确认。直至显示如图的硬件仿真确认对话框。



我们现在用硬件仿真方式来调试这个程序，因为程序是对 P1 端口加 1 操作，我们可以打开外设的端口来观察 P1 口。方法是选择主菜单[**外设 | 端口**]功能打开端口窗口。重新编译程序，全速执行程序，因为有断点，程序会暂停在断点处。我们观察端口窗口的 P1 口值，会发生变化。再次全速执行，观察 P1 口的变化。同时也可以用电压表去测量仿真头的 P1 管脚，可以看到 P1 管脚也随之发生变化。点击端口窗口的 P1 口的白框来改变 P1 口的值，再次运行程序，P1 从改变后的值加 1。（P1 口的值也可以从 SFR 窗口观察、修改）

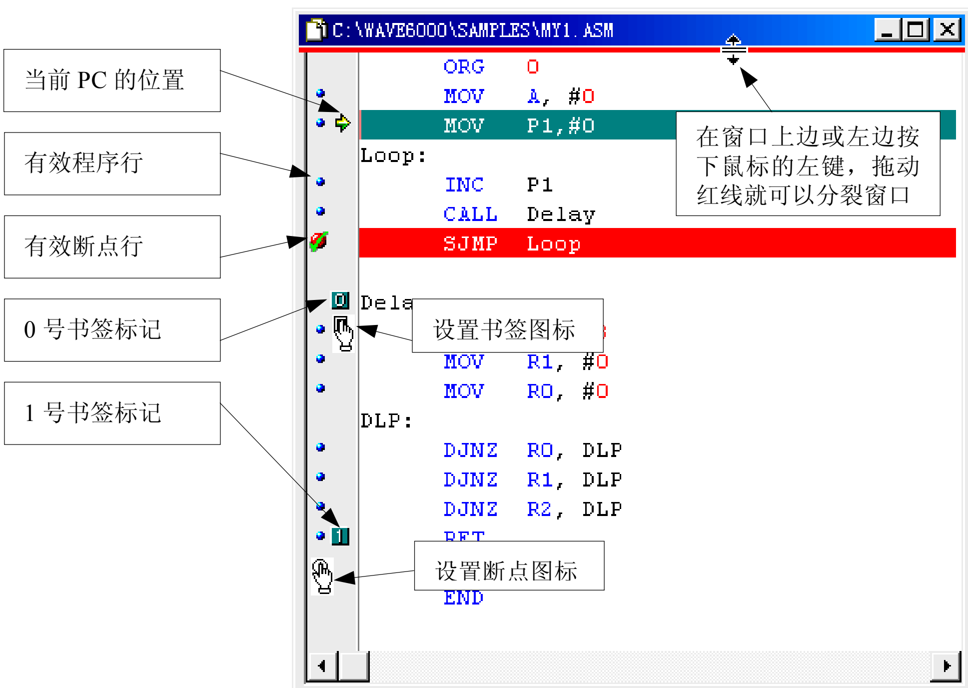


如果用户已经有写好的程序，可以从第 3 步“新建项目”开始，将你的程序加入项目，就能以项目方式仿真了。如果用户不想以项目方式仿真，则要先关闭项目，再打开你的程序，并且要正确设置仿真器、仿真头，然后再编译、调试程序。

到此为止，你已经学会使用伟福的仿真环境了。在使用过程中，你会逐步提高自己的技能。伟福仿真器的更多功能可参考本说明书的其它部分。

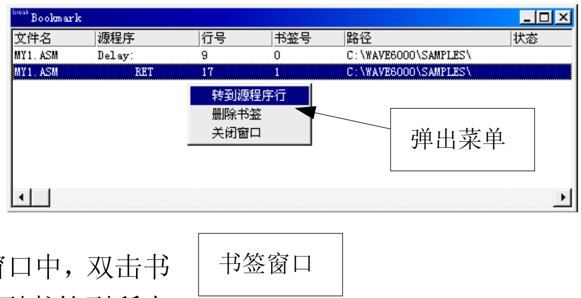
伟福文本编辑器的使用

伟福文本编辑器用来输入程序，使用方便。具有与 C 语言、汇编语言、PLM 语言语法相关的彩色显示，使编写程序更加轻松，观察程序醒目。并且用户可按照自己的喜好自己设置颜色，享受个性化编程带来的乐趣。可以在编辑窗口中设置断点、书签，用于快速定位程序，对于编写、分析、比较、检查较长的、复杂的程序非常有帮助。查找功能可以在程序中查找、替换字串。在编辑窗口中，可以查找配对符号，如找到‘{’相对的‘}’或找到与‘(’相对‘)’，并且将中间的部分加亮显示，这样在复杂的嵌套中确定程序的块结构。可以在编辑窗口中对多行程序同进同退，帮助您编写优美、整洁的程序。窗口分隔功能可将源程序窗口分成两个或三个完全独立的编辑窗口，而所编辑的内容却是同一程序，为分析、比较检查大程序提供方便。



设置断点、书签

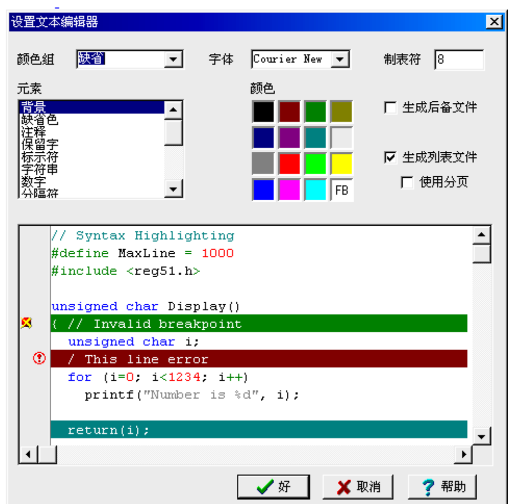
编辑窗口的左边界用于显示断点、书签。将鼠标移到边界的右半边，光标变成“手及方块”此时单击鼠标左键就可以设置书签，共可以设置多达 10 个书签，标号从 0 到 9，如果当前位置上已有书签，单击鼠标则去除此书签，书签的添加、删除操作也可以在书签窗口中实现。在书签窗口中，双击书签号就可以将编辑窗口中的光标快速定位到书签到所在的行。此功能对于在大程序迅速定位很有帮助。移到左半边则可以设置断点。



将光标移到编辑窗口边界的左半边，光标变成“手及圈”，单击鼠标左键设置断点，也可以用弹出菜单的“设置/取消断点”功能或用 Ctrl+F8 组合键设置断点。如果不在调试程序，断点图标为“红圆”，在调试程序时，如果断点有效图标为“红圆绿勾”，若当前行已有断点，单击左键就会删除此断点。同样断点的添加、删除也可以在断点窗口中操作。断点除了在调试时让程序暂停的功能外，断点也可以象书签一样，可以快速定位程序的位置。

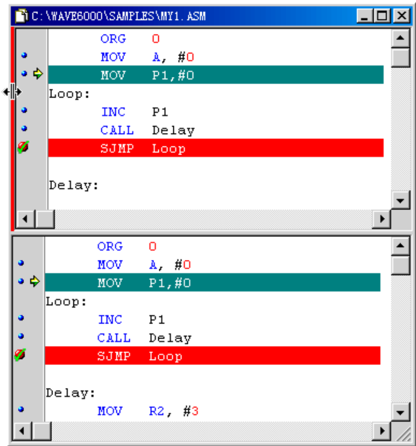
设置个性化编辑器颜色

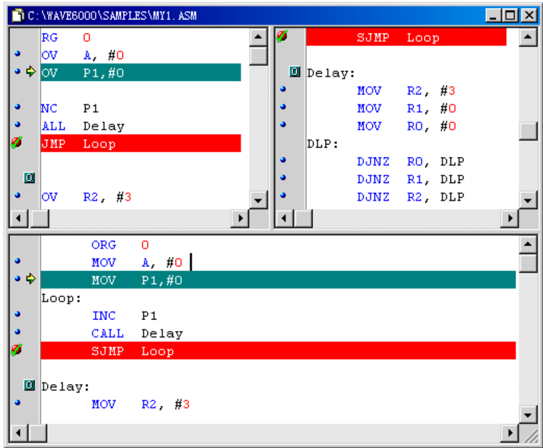
选择主菜单的[仿真器 | 设置文本编辑器]功能，就可对编辑器的颜色进行设置。软件环境中已经预定义了四组颜色，为“缺省”、“古典”、“高亮”、“海洋”。用户如果对这四种都不满意，可以自己定义自己喜爱的颜色，在“元素”栏中选择要定义的元素，在“颜色”栏中定义背景色和前景色。在“字体”栏中选择字体。定义颜色和字体后，可以下面的图例中看到编辑器产生的效果。



分裂多窗口

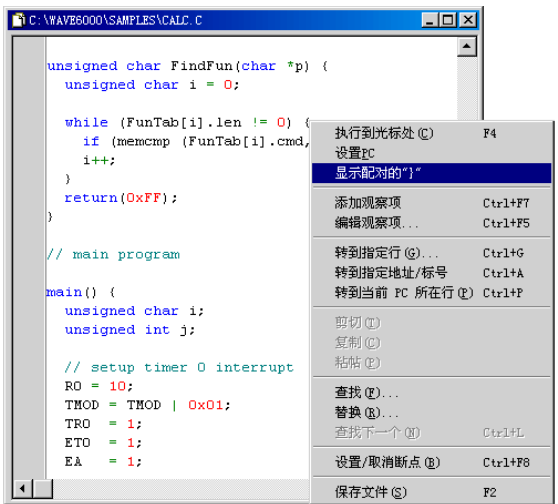
源程序编辑窗口可以分割成两个或三个窗口，用于观察同一程序的不同位置。各个分窗口的横竖滚动棒可以独立控制。在编辑窗口的上方按下鼠标左键，就会出现一条红线，表示窗口分割线，拖动红线大于一定距离松开，就可以分裂窗口。若想关闭分窗口，在窗口分界线上按下鼠标左键，也会出现红线，拖动到上/下边小于一定距离松开，就会关闭分窗口。若相再分出一个窗口，可在窗口左边上方按下鼠标左键（如左图），拖动红线可分出第三个窗口（如右图）。





查找匹配符号

在编写多重嵌套结构的程序时，有时非常想知道某一符号对应的符号在何处，这用到了查找匹配符号的功能。当光标放在要查找的符号上，按鼠标右键弹出菜单，选择“查找匹配符号”功能，两配对符号之间的程序会被加亮显示



多行文本同进同退

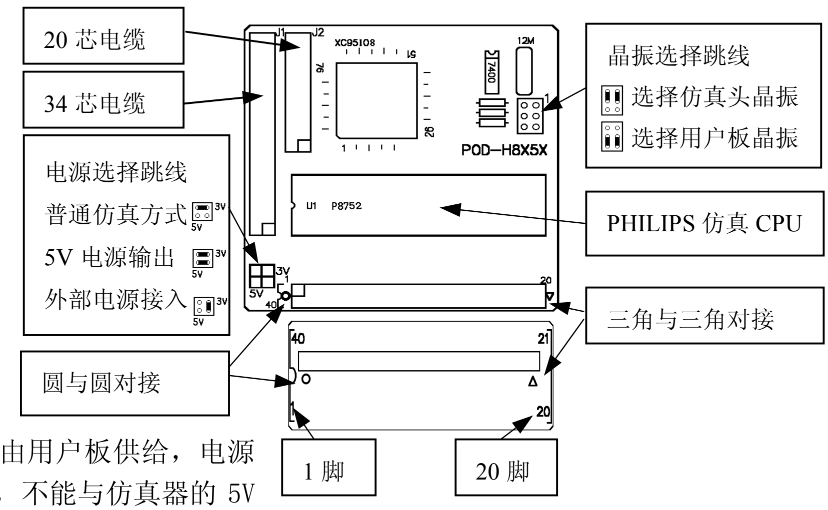
用多行文本同进同退功能，可以将程序排列整齐，首先选中需要排列的程序块，按住 Ctrl 键，按“U”键可将文本左移一位，若按“I”键会将文本右移一位。

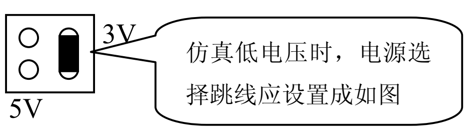
PODH8X5X 使用说明

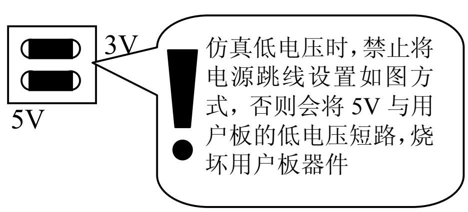
PHILIPS 公司生产的单片机都带有一种被称为 Hooks 的技术，此技术可用来对本芯片进行仿真，PODH8X5X 就是利用 PHILIPS 公司授权的 Hooks 技术设计的。PODH8X5X 的特点是：当你需要仿真某种单片机时，只要将该芯片换到仿真头上即可，而无须更换仿真器或仿真头，这些芯片包括 P87C5x、P89C51Rx、P89C51Rx+、P89C51Rx2、P87C51FA/FB、P89C66x 等，注意，P87C591 的控制时序与前几种略有不同，不能直接将 P87C591 换到 PODH8X5X 仿真头上来仿真，要用 PODH591 仿真头来仿真。既然是用芯片仿真自己，那么芯片固有的特性就可以在仿真时体现出来，例如低电压工作、6 时钟/12 时钟、内部扩展 RAM，还有芯片上的 A/D 变换功能（P87C591）、I2C 总线等功能都能仿真。**注意：PODH8X5X 仿真头上原配的芯片为 P87C52，用于仿真通用的 8X5X 系列单片机，用户需要仿真其它类芯片，如 RD、RD+、RD2、66X 时，必须要将仿真头的芯片换成该类芯片。用于仿真头的芯片内部不能有程序，如有程序，应先将其内部程序擦空，6/12 时钟的配置也在同时写片。**

1. 仿真低电压

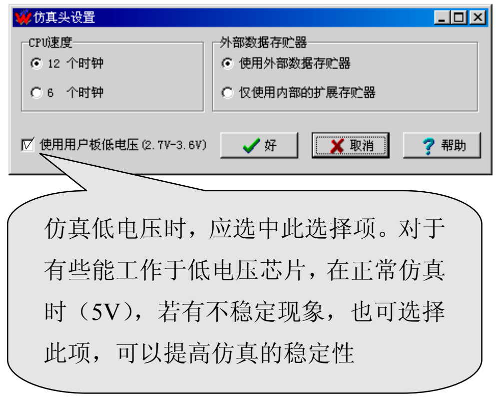
当用户需要仿真低电压时，要注意几点：1）仿真头的仿真芯片必须能工作于低电压状态。2）有的芯片工作在低电压时，晶振频率有限制，具体的工作电压与频率的关系应参见该芯片的资料。3）仿真低电压时，仿真头的电源由用户板供给，电源选择跳线应设置正确，不能与仿真器的 5V 电源短路。





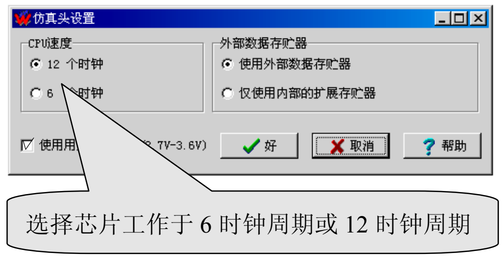


4）在软件中，仿真头设置时，应选中“使用用户板低电压(2.7V-3.6V)”选择项。操作步骤如下：在主菜单上选择[仿真器 | 仿真器设置]，在“仿真器设置”栏中选择好 PODH8X5X 后，再按“仿真头设置”按钮，出现如图对话框。由于各类芯片的特性不同，在正常仿真（5V）时，若有工作不稳定的现象，也可以选中此项，可能会提高仿真的稳定性。



2. 选择 6/12 时钟

PHILIPS 公司的很多芯片可以工作在 6 时钟/周期和 12 时钟/周期两种速率，这要求在编程时设定，在仿真时，也应该将用于仿真的单片机的工作时钟设置成要求的速率，用编程器烧到仿真芯片中，使其能工作在相应的时钟频率下。例如，当你选择 6 时钟/周期时，应先对仿真头上的仿真 CPU 进行编程，擦除里面的程序，将芯片的工作时钟烧成 6 时钟/周期。软件环境的也要做相应地设置。PHILIPS 有一些芯片缺省工作方式为 12 时钟/周期，有一些芯片为 6 时钟/周期，要注意的是：有的芯片时钟选择位只能编程一次，不可恢复，在使用前请阅读芯片资料。



3. 访问 XDATA 方式的设置

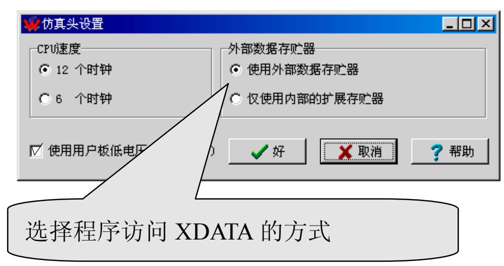
有的芯片内部带有扩展 RAM，象 P89C51Rx2，P89C66x 等芯片，这样这些单片机既可访问外部的 XDATA，也可访问内部的 XDATA，访问方式的设置如图。当仅使用芯片内部的扩展 RAM 时，请选择[**外部数据存贮器**]下的 “仅使用内部的扩展存贮器”，这样打开外部数据（XDATA）窗口观察外部数据时，就不会影响到 P0/P2 端口。注意：有的芯片在复位后，缺省状态就是访问内部 XDATA，例如 RD2，而有的芯片复位后的状态是访问片外 XDATA，例如 P89C66x。建议在程序中用指令去设定访问方式，而不要直接用缺省方式。

[**使用外部数据存贮器**]

用户在程序中使用 MOVX 指令，

P0,P2 口在执行非 MOVX 指令时作 I/O 用

P0,P2 口在执行 MOVX 指令时作总线用



[**仅使用内部的扩展存贮器**]

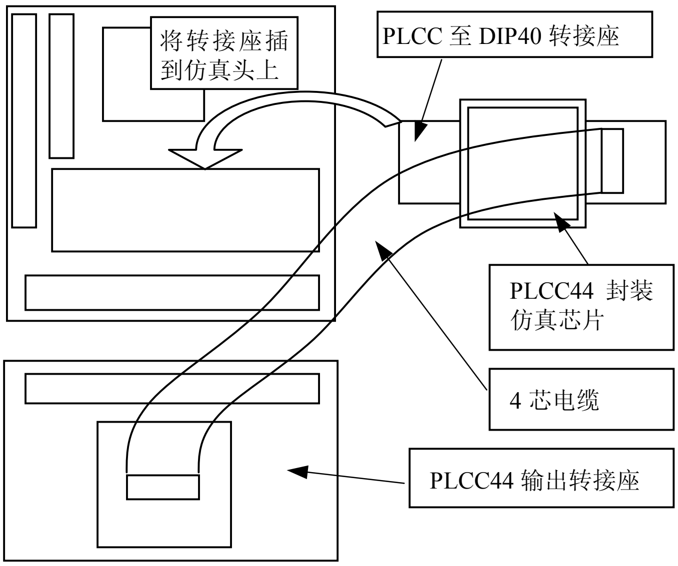
P0,P2 口在执行所有指令时均作 I/O 用，这种方式主要用作内部含 XDATA 存贮器的单片机仿真（如：P89C66x、P89C51RD2）

用 P89C66X、P89C51RD2 等内部带扩展 XDATA 的芯片除可仿真同类芯片外，还可以用

来仿真 W78E58B，W78E516，SST89C58 等内部含 XDATA 的单片机，但请注意，各家单片机的内部扩展 XDATA 的允许位是不同的，在仿真时要按 66X 或 RD2 的方式设置，而在仿真完成后，编程到具体单片机之前要做对应的改变，请参阅各家芯片的资料。

4. 使用 PLCC 转接座

有些芯片为 PLCC44 封装，如 P89C66X，P87C591 等，在用 PODH8X5X 仿真这类封装芯片时，要用 PLCC44 至 DIP40 的转接座，将芯片转成 DIP40 方式，然后再插到仿真头上，仿真输出时，还要用 PLCC44 的输出转接座插到用户板上。若 PLCC44 比 DIP40 多出的 4 脚为功能脚（不为空脚），例如 P87C591，可以用电缆将 4 脚接到输出转接座上。



5. 注意事项

在使用 HOOKS 仿真头时，有些状态的改变会影响仿真功能，所以有几点请注意：

1）在对 AUX 改变时，不要使用 MOV AUX，#XX 指令，这样会改变寄存器其它位，而这些可能是不可以改变的，若需要对某位操作，请使用 ANL，ORL 来清或置相应位。

2）在仿真时，不可关闭 ALE 输出。

3）在仿真时，应关闭看门狗功能。

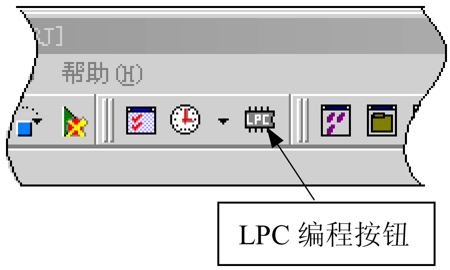
另请注意，在用外晶振时（用户板晶振），若晶体振荡不好，可以将用户板上晶振边的电容去掉，可以改善振荡性能。

LPC 编程器使用

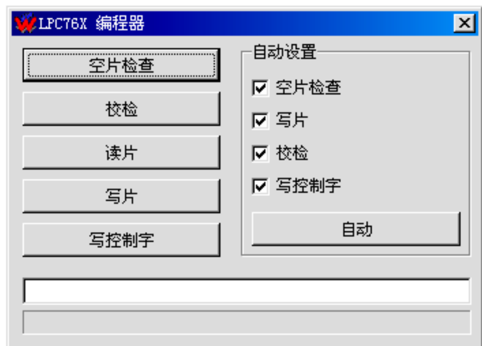
在伟福开发环境中，使用 LPC76X 或 LPC93X 的编程器有两种方法

一、仿真完成后，直接编程。

当用户使用 LPC 仿真头仿真程序无误后，将仿真器的电源线和串行通信电缆接到 LPC 编程器上。按 LPC 编程器按钮。注意：**在编程之前一定要确认程序编译正确无误。**



在下图的编程对话框中，左边的按钮可以完成独立的功能。可以执行“空片检查”、“校验”、“读片”、“写片”、“写控制字”各单项功能。“自动”按钮用来连续执行被选中的功能，完整地执行一次芯片编程操作。

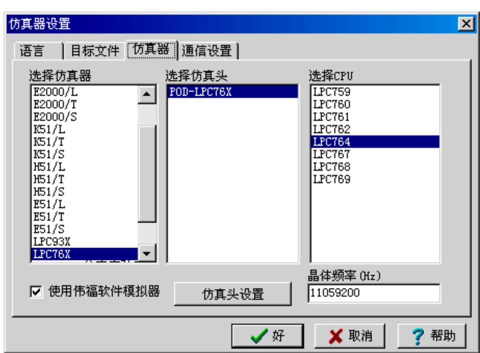


若要对芯片批量重复编程，只要在芯片编程完成后，看到电源灯慢速闪动，表示可以再次编程。换上空芯片后，按下编程器上的“YES”按钮就可以再次对芯片进行编程。

二、装入目标文件进行编程。

如果程序是已经开发好的目标代码，可以用装入目标文件方式进行编程。

1．在菜单“仿真器设置”中选择仿真器/仿真头为 LPC76X 或 LPC93X

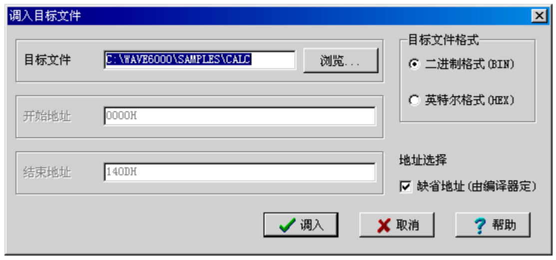




2．在通信设置中，选择正确的串行通信口，再将仿真方式设为“使用伟福软件模拟”

3．在文件菜单中选择“调入目标文件...”功能

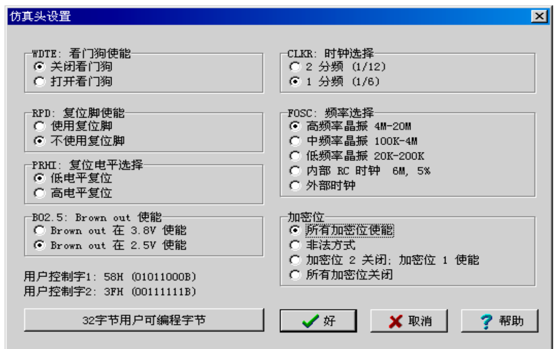
输入目标文件名，确定正确的目标文件格式（BIN 格式还是 HEX 格式），地址选择为“缺省地址”。



4．设置 LPC 的控制字。

按照设计要求，设置好芯片的控制字

LPC76X 的控制字设置窗口



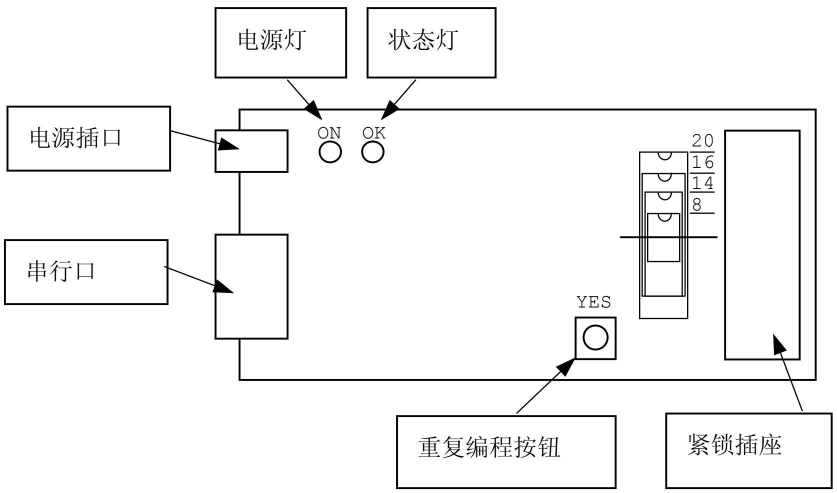


LPC93X 的控制字设置窗口

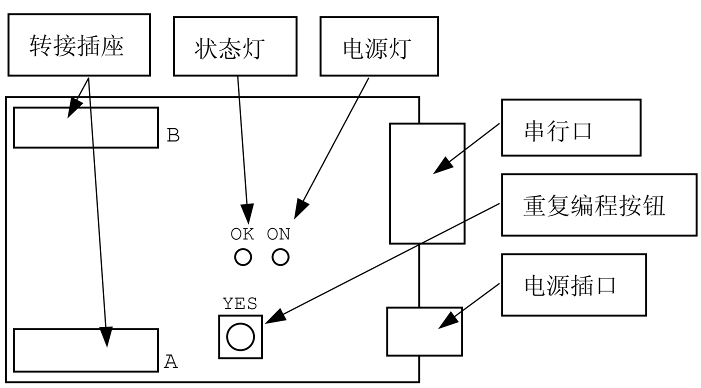
5．按照第一步“直接编程”的步骤，接好电源和串行电缆，按 LPC 编程按钮。然后再选择“自动”，完成芯片编程。

三、LPC 编程器的硬件

LPC 编程器的电源和串行口电缆与仿真器相同，电源灯正常状态为常亮，在编程器与计算机通信时，电源灯会快速闪动。当一片芯片编程完成后，电源灯会慢速闪动，表示可以再次编程，这时换上一片芯片，按“YES”键就可以重复编程，适合批量生产时用。状态灯在编程过程中会亮，表示状态为忙。

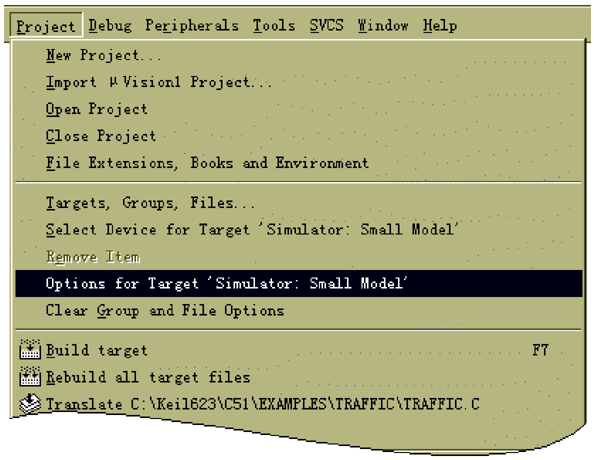


LPC76X 编程器



LPC93X 编程器

在 Keil 的 uV2 集成环境下使用伟福仿真器。



若想在 Keil 的 uV2 集成调试环境下使用伟福仿真器。需要在 Keil 环境中做如下设定：

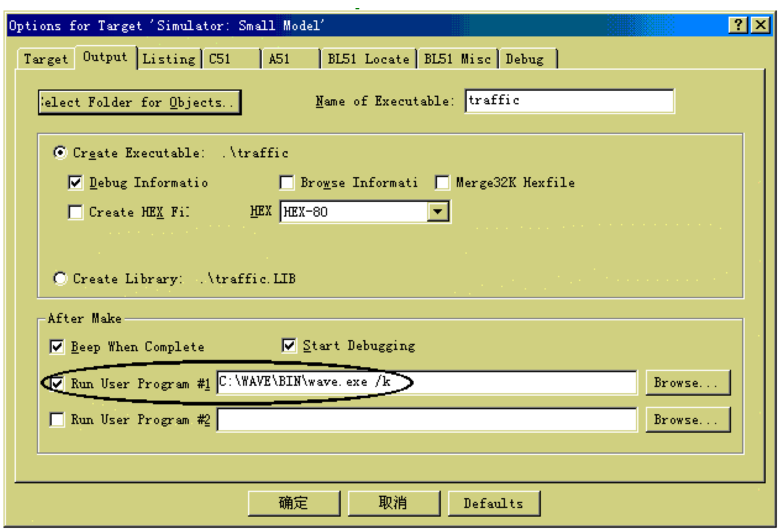
A. 设置“编译后运行程序”，以便初始化伟福仿真器。

B. 设置硬件调试方式，并设置串口参数

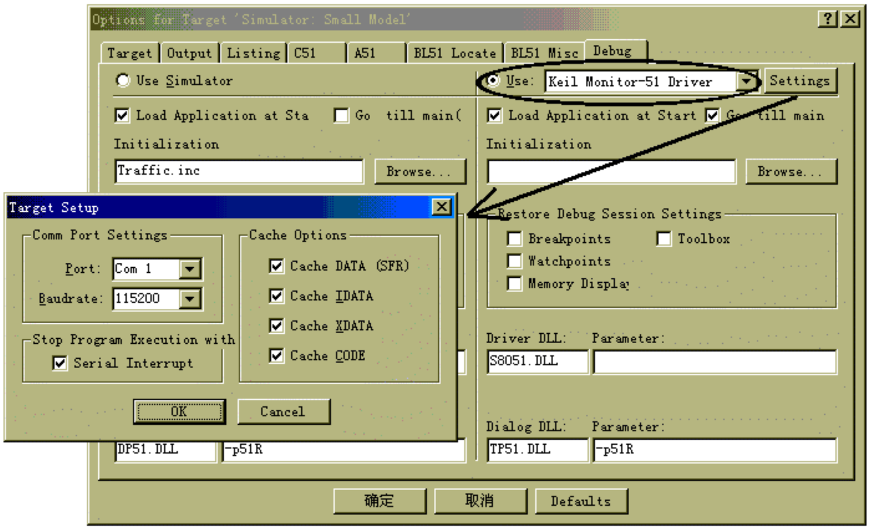
具体步骤如下：

1. 在 Keil 环境下建立项目，并把程序做为模块加入项目。

2. 在主菜单上，选[Project]->[Options for Target ‘...’]，如图



3. 出现如图的设置窗，选择[Output]栏，选中[Run User Program #1],使其有效。在随后的文字框中填入“C:\WAVE\BIN\WAVE.EXE /K”,其中“C:\WAVE\BIN\”是伟福开发环境安装路径，“WAVE.EXE /K”是要求 WAVE.EXE 执行完初始化后退出，以便在 Keil 环境中用伟福仿真器来仿真。
4. 在设置窗中，再选择[Debug]栏，选中[Use Keil Monitor-51 Driver]，按随后的[Settings]按钮，出现“Target Setup”对话框，选择正确的通信口及波特率，选中其它所有项。设置后，按确定退出。(波特率为 115200)



这样就可以用伟福仿真器在 Keil 环境下工作了。程序写完后，如果编译没有错，就会出现伟福的仿真器设置对话框，选择正确的仿真器、仿真头和仿真 CPU 后，设置正确无误后，伟福环境会自动退出。

如何用 PODPIC67XP 仿真 PIC16C711 芯片

用 POD-PIC67XP 仿真头仿真 PIC1671/711 芯片时。所有寄存器及 RAM 开始地址,先按 PIC16C73/74 芯片的地址定义,仿真完成后,再按 PIC16C71/711 的地址及 RAM 开始地址重新定义,编译生成程序,烧到芯片中。主要是 ADCON 及 RAM 开始地址。本例子用 POD-PIC711 仿真头和 POD-PIC67X 仿真头都试过,通过改变编译控制就能产生正确的机器码，都能正常运行。

	寄存器	PIC16C72/73/74	PIC16C710/711
AD 控制寄存器 0	ADCON0	1FH	08H
AD 控制寄存器 1	ADCON1	9FH	88H
AD 转换结果寄存器	ADRES	1EH	09H/89H
AD 转换中断标志	ADIF	0CH.6(PIR1.ADIF)	08H.1(ADCON0.ADIF)
AD 转换中断允许	ADIE	8CH.6(PIE1.ADIE)	0BH/8BH.6(INTCON.ADIE)
RAM 开始		20H	0CH

```
;
;若仿真头为 POD-PIC67XP 就用 “POD equ 74” 来定义寄存器组
;若仿真头为 POD-PIC711 就用 “POD equ 71” 来定义寄存器组
;因为这里寄存器已定义，在语言设置时，请不要选中“伟福预定义符号”
;
;POD     equ 71    不是使用 POD-PIC711 仿真头
POD     equ 74  ;选择使用 POD-PIC67XP 仿真头

if POD=74    ;根据 POD 的定义来选择编译控制
  ;选用 PIC67X 仿真头  PIC16C74
  PA_D    EQU 6    ;PORTA 工作于数字输入/出
  PA_A    EQU 4    ;PORTA.0 工作于模拟输入
  T0IF    equ 2
  GO      equ 2
  ADIF    equ 6
  PIR1    EQU 0CH
  ADRESH  equ 1eh
  ADCON0  EQU 1FH
  ADCON1  EQU 9FH
  PIE1    EQU 8CH
else
  ;选用 PIC71X 仿真头  PIC16C711
  PA_D    EQU 3    ;PPORTA 工作于数字输入/出
  PA_A    EQU 2    ;PORTA.0 工作于模拟输入
  T0IF    equ 2
  GO      equ 2
  ADIF    equ 1
  ADRESH  equ 09h
  ADCON0  EQU 08H
  ADCON1  EQU 88H
  PIR1    EQU ADCON0   ;在 711CPU 并无 PIR1 寄存器
  PIE1    EQU 8BH      ;在 711CPU 并无 PIE1 寄存器
endif

ADIE    EQU 6
STATUS  EQU 03H
PORTA   EQU 05H
```

```
PORTB   EQU 06H
PORTC   equ 07h
INTCON  EQU 0BH
OP_REG  EQU 81H
TRISA   EQU 85H
TRISB   EQU 86H
trisc   equ 87h

        ORG     0H
iostart:
        BSF     STATUS,5
        MOVLW   PA_D
        ;========^^此处也根据芯片不同而不同
        ;==在 16C72/73/73A/74/74A 中，03H 是将 A 口设为模拟输入口。
        ;==而在 16C71/710/711 中，03H 才是将 A 口设为数字口。
        ;==另请注意：A 口的第四位(PORTA,4)为输出时为 OC 门，
        ;==要加上拉电阻才能为高。详细资料请从 MICROCHIP 下载

        MOVWF   ADCON1
        MOVLW   0H
        MOVWF   TRISA
        BCF     STATUS,5
        MOVLW   13H         ;测 PA 口数字方式输出
        MOVWF   PORTA
        MOVLW   10H
        MOVWF   PORTA
        NOP
adstart:                    ;测试 A/D
        clrf    PORTB       ;清除 PORTB
        movlw   10000001b   ;A/D 转换时钟为 Fosc/32， 允许 A/D
        movwf   ADCON0
        bsf     status,5
        movlw   03fh
        movwf   TRISA       ;PORTA 输入
        movlw   10000111b   ;TMR0 预置， 1:256
        movwf   OP_REG
        clrf    TRISB       ;PORTB 输出
        movlw   PA_A        ;Vref=VDD,A0/1/3 为模拟输入
        movwf   ADCON1      ;

        bcf     status, 5
Main:   btfss   INTCON,T0IF     ;Timer0 定时等待
        goto    Main
        bcf     INTCON,T0IF
        bsf     ADCON0,GO       ;启动 A/D 转换
Wait:   btfss   PIR1,ADIF       ;等待转换结束
        goto    Wait
        bcf     PIR1,ADIF
        movf    ADRESH, 0       ;将转换结果送 PORTB
        movwf   PORTB
        goto    Main            ;循环

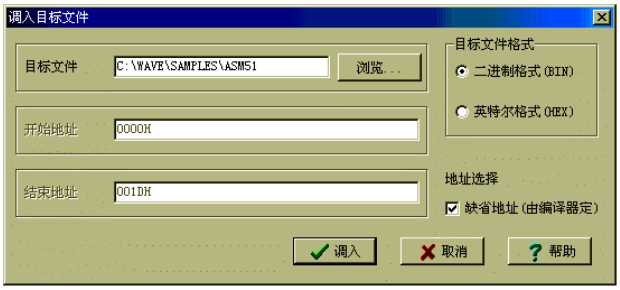
        END
```

使用伟福软件的反汇编功能

伟福软件的反汇编功能强大，既能生成带地址的汇编代码，也可以生成汇编源程序。如果运用独创的反汇编控制方式，就可以生成可读性极强的源程序。汇编步骤：

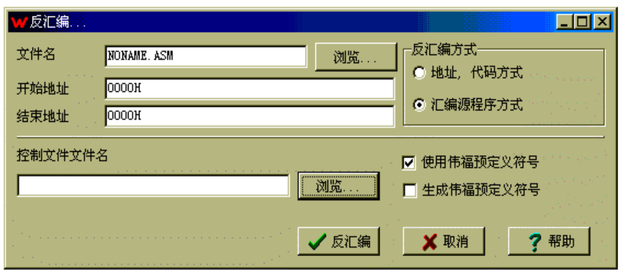
1. 调入目标文件

在菜单中选[文件]->[调入目标文件...]，出现如图对话框，用[浏览]选择所要反汇编的目标文件，指定正确的文件格式（二进制格式/HEX 格式），地址选择最好用“缺省地址”。调入文件后，最好打开 CPU 窗口，查看一下窗口中的程序是否正确。



2. 反汇编程序(汇编源程序方式,无控制文件)

在菜单中选[文件]->[反汇编...]，出现如图对话框，填入所要生成的汇编源程序文件名，反汇编的开始/结束地址可以用缺省指定的，也可以指定程序中的某段程序来反汇编。反汇编方式选用“汇编源程序方式”，如果是第一次反汇编，可以不用控制文件。在伟福仿真软件内部已经定义了常用的寄存器的名称，如果选用“使用伟福预定义符号”，在反汇编时，会自动使用这些符号，否则源程序中会以地址来表示相应的寄存器。如果选用“生成伟福预定义符号”，会在源程序中定义我们内部符号，以便程序可以在其它环境中编译。



3. 观察生成的程序,判断程序区,数据区，无用数据等



反汇编完成后,会在信息窗口中有所显示，如果正确，双击反汇编生成的文件名，可以打开生成的源程序，由第一条指令开始，分析程序的结构，判断程

序中的数据区，代码区，无用数据区，最好做个记录，便于下一步建立控制文件使用。

4. 建立反汇编控制文件(格式如下),保存控制文件
新建一个文本文件，按照反汇编控制文件格式（格式见随后说明），输入控制文件，存盘。

5. 调入目标文件
按第 1 步的方法，再次调入目标文件。

6. 反汇编程序(程序方式,有控制文件)
参照第 2 步的方法，反汇编程序，此时与前面不同的是，这次有控制文件，要在[控制文件名]里填入第 4 步生成的控制文件名。让反汇编程序按照此控制文件生成反汇编源程序。

7. 进一步分析生成的程序,找出更详细的程序功能,数据区
重新打开生成的源程序，按第 3 步的方法，进一步分析程序，找出更详细的程序功能，数据区，并做记录。

8. 修改反汇编控制文件,存盘.
按再次分析的结果，修改反汇编控制文件，存盘。

9. 重复 第 5 步至第 8 步, 直到生成详细程序.
不断地重复上述第 5 步至第 8 步，一步一步的细化程序，直到生成详细程序。

反汇编控制字使用说明：

1. REM　注释，相当于程序中的说明语句
   格式: REM 注释语句
2. CODE　标号定义，将代码中某个地址定义为一个有意义的符号
   格式: CODE（'标号名称', 地址, '说明语句'）
3. DATA　数据区标号定义，将数据中的某个地址定义为符号
   格式: DATA（'数据标号名称', 地址, '说明语句'）
4. XDATA　外部数据标号定义(51 系列)
   格式: XDATA（'外部数据标号名称', 地址, '说明语句'）
5. BIT　位标号定义(51 系列),
   格式: BIT（'位标号名称', 地址, '说明语句'）

6. DB　程序区数据字节定义,将数据定义成字节形式
   格式: DB　('数据标号', 开始地址-结束地址, '说明语句')
7. DW 程序区数据双字节定义,将数据定义成双字节形式
   格式: DW　('数据标号', 开始地址-结束地址, '说明语句')
8. DS　程序区数据字串定义,将数据定义成字串形式
   格式: DS　('数据标号', 开始地址-结束地址, '说明语句')
9. NULL 无效数据区,反汇编时,跳过这段数据
   格式: NULL (开始地址-结束地址)

以下是一个反汇编控制文件的例子:

```
code  ('START', 1ADh, '======= This is MAIN')
code  ('Read_RX',30h, '======= test if sio in')
code  ('main_loop',1eah,'======= here waiting command')
code  ('Wait_RX',4eh,'')
code  ('Process_11H',1fbh,'======Function 11H')
code  ('Process_12H',228h,'======Function 12H')
rem code  ('addr132e',132eh,'')
rem code  ('JUMPTBL0',1462h,'=========== function jump')
bit   ('Has_RX',73h,'')
db    ('NoUse', 3h-0fh)
db    ('xx1Table',0b8h-0c7h)
rem db    ('xx3Table',09c1fh-9c3ch)
null  (136ch-1fffh)
```

5 分析功能使用

影子存储器

所谓影子存储器就是在仿真环境中为外部存储器建立一个影子,可以在程序运行时,动态地观察外部存储器的状态,通过影子存储器不但可以看到程序运行时,外部存储器的值,还可以观察到外部存储器是否被存取过. XDATA 窗口和观察窗口, 约每 2 秒刷新一次,这样就可以看到存储器当前值.为程序动态调试提供了更有效的手段.在程序运行时,XDATA 窗口及观察窗口中数据背景变为橄榄绿色,表示相应数据被访问过.这样可以方便地观察在上次程序运行过程中有哪些变量发生了变化.

以 51 为例,说明影子存储器的使用方法.

1.输入以下程序并存盘:

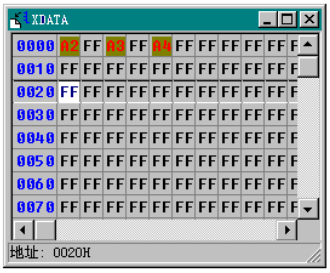
```
      mov   a, #0
loop: mov   dptr, #0
      movx  @dptr, a
      mov   dptr, #2
      inc   a
      movx  @dptr, a
      mov   dptr, #4
      inc   a
      movx  @dptr, a
      call  delay
      sjmp  loop

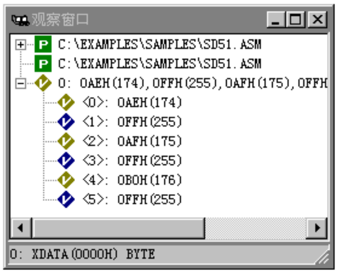
delay:mov   r5, #10
      mov   r6, #0
      mov   r7, #0
dlp:  djnz  r7, dlp
      djnz  r6, dlp
      djnz  r5, dlp
      ret
      end
```

2. 新建项目,并将上面文件做为模块加入
3. 编译,调试
4.选择分析功能为[影子存储器]
5. 打开 XDATA 窗口,打开观察窗口
6.在观察窗口中加入观察项,以观察 XDATA 的地址 0000 处的数据,

方法是:　　在'**加入观察项**'窗口中,[**表达式**]:0,[**重复次数**]:6
[**显示格式**]:'混合十,十六进制',去掉[**缺省显示方式**]前的选择勾
[**存贮区域**]:'XDATA', 选中'**存贮器内容**'.

7.运行程序,就可以在 XDATA 窗口和观察窗口约每两秒刷新一次,
同时被观察的数据也随之改变.

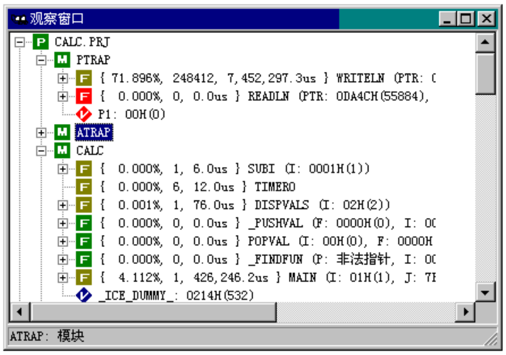




0,2,4 号单元显示颜色为橄榄绿,表明该单元被访问过(读或写过)
0,2,4 号单元显示的值为当前该单元的值,XDATA,观察窗口被不断刷新.

程序时效分析

分析程序中,各过程,函数以及每条指令的运行时间, 执行次数及占整个程序运行时间的百分比,了解程序执行效率,可以优化程序,进一步改善程序性能.



1. 橄榄绿色表示此函数已经执行过,红色表示函数中某个变量被改变.
2. {X%,Y,Zus}大括号中表示程序时效分析内容,有三种数字,X 表示此函数的执行时间占总执行时间的百分比 Y 表示此函数被执行的次数,Z 表示此函数总的执行时间.

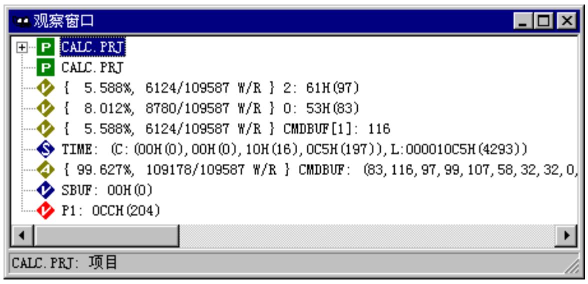
{71.896%, 248412, 7,452,297.3us} WRITELN 表示:
PTRACP 模块中的 WRITELN 函数执行了 248412 次, 运行时间为 7,452,297.3us,
占总执行时间的 71.896%
{0.000%, 6, 12.0us} TIMER0 表示:
CALC 模块中的 TIMER0 函数 执行了 6 次, 运行时间为 12.0us, 占用 0.000% 执行时间(占用%太小, 显示不出来)因为 TIMER0 为中断服务程序, 这也说明发生了 6 次中断.

如果你在观察窗口中加入标号, 你就可以在观窗口中看到该标号的执行次数.
例如你在程序中有以下产生脉冲的语句:

```
Output:  SETB     P1.0
         CLR      P1.0
```

现在你想知道你的程序输出了多少个脉冲, 你就可以选择程序时效分析, 将 Output 加入观察窗口,全速执行程序, 在程序暂停后, 在观察言窗口就可以看到标号 Output 的执行次数, 也就是产生脉冲的个数. 同样, 你也可以将中断入口的标号加入观察窗口, 这样你就可以知道中断函数的执行次数.

数据时效分析

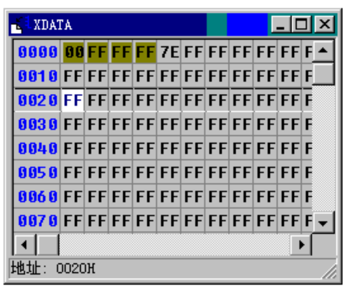


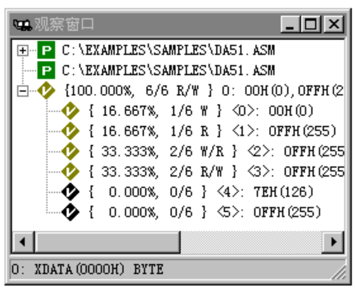
数据时效分析程序中,可以统计出各变量,数据单元是否被访问及被访问的次数,访问频率,和访问方式从而改善程序的结构.开发出更有效,更稳定的程序.

1. 橄榄绿色表示此变量被访问过过,红色表示变量在上次执行过程中被改变.
2. {X%, Y/Z, W/R},大括号中为数据时效分析内容,X 为此变量被访问次数占总访问次数的百分比,Y 为此变量被访问次数,Z 为总的访问次数(所有访问以字节计数,若变量类型为字,以两倍计数,若变量为长整形,则以四倍计数),W 表示变量被写过,R 表示变量被读过,若只有 W 或 R,表示此变量只被写或读过. 若 R 在先,W 在后,表示变量被先读后写,先 W 后 R,表示变量被先写后读.

1. 输入下列程序, 该程序访问 0,1,2,3 号单元.
2. 在观察窗口加入: 表达式: 0　　重复次数: 6　　缺省方式显式: 不选择　存贮区域: XDATA
   即从 0 号单元开始, 显示 6 个 XDATA 区域单元内容.
3. 打开 XDATA 窗口, 观察窗口. 全速执行到第 12 行. 你可以看到下列结果

```
mov    a, #0
mov    dptr, #0
movx   @dptr, a      ; 写 0 号单元
mov    dptr, #1
movx   a, @dptr      ; 读 1 号单元
mov    dptr, #2
movx   @dptr, a
movx   a, @dptr      ; 写/读 2 号单元
mov    dptr, #3
movx   a, @dptr
movx   @dptr, a      ; 读/写 3 号单元
sjmp   $             ; 全速执行到这行
end
```





0,1,2,3 号单元显示颜色为橄榄绿,表明该单元被访问过(读或写过),{ }内为单元的访问状况

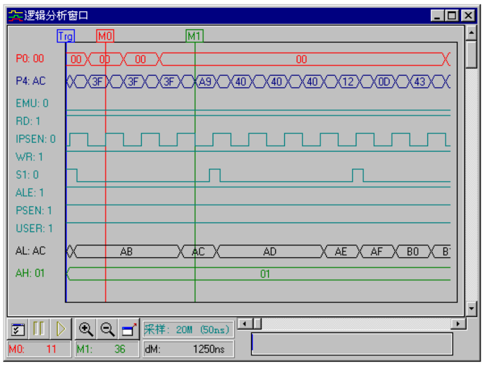
{16.667%, 1/6 W} 表示 XDATA 共被访问过 6 次, 0 号单元被写一次,占访问次数的 16.667% 0 号单元应是一个 I/O 输出口, 一个变量不应仅写不读.

{16.667%, 1/6 R} 表示 1 号单元被读这一次, 1 号单元应是一个 I/O 输入口, 一个变量不应仅读不写.

{33.333%, 2/6, W/R} 表示 2 号单元被写/读过且写在前,读在后.

{33.333%, 2/6, R/W} 表示 3 号单元被读/写过且读在前,写在后. 这里可能隐含未初始化错误,因为一个变量不应在写前被读.

逻辑分析仪



逻辑分析仪是一种类似于示波器的波形测试设备,它可以监测硬件电路工作时的逻辑电平(高或低),并加以存储,用图形的方式直观地表达出来,便于用户检测,分析电路设计(硬件设计和软件设计)中的错误,逻辑分析仪是设计中不可缺少的设备,通过它,可以迅速地定位错误,解决问题,达到事半功倍的效果.

功能强大的 E6000/L 逻辑分析仪可以同时采样 40 路波形,并且每路采样深度为 32K,采样速度可达 20MHz,可以外接 16 路,以观察分析用户板上电路工作情况.更有强大事件触发功能,用户可以指定在何种条件下,对电路进行采样分析.并且可以指定,在事件发生前/中/后三种位置上采样,以便精确地捕捉到事件发生时,电路工作的状态.为软硬件设计提供强有力的帮助.波形共有 40 路,分五组,分别为 A 组数据线,B 组数据线,C 组控制线,D 组地址线低八位,E 组地址线高八位(对 E51/L 型仿真器,只有一组外接线).显示的波形除这 40 路外,还有一组用于显示备份信号 F0,便于用上次备份的信号与此次采样信号相比较,参见弹出菜单的“备份”功能.如果外接探头,可以接到仿真器的 J3 或 J4 插槽,在设置时选中 J3 或 J4.由于逻辑分析仪与仿真器相互独立(K51/L 和 H51/L 型仿真器除外,K51/L 及 H51/L 的逻辑分析仪随程序运行而开始,程序停止就停止),所以既使用户不在仿真状态,只要加上电,逻辑分析仪就可以工作。

不同的 CPU,在逻辑分析仪窗口中显示的信号名的所不同.最后三组 AL 组为地址低八位,AH 组为地址高八位,F0 组为备份信号. 前三组根据 CPU 类型不同,定义如下:

☆对于 POD-8X5XP 仿真头:

P0 组:当被仿真 CPU 为 805X 时,P0 为数据总线

P4 组:当被仿真 CPU 为 875X 时,P4 为数据总线.

EMU: 当 EMU 为高时,表示在监控状态,为低时表示在用户状态.
RD: CPU 的读信号
IPSEN:当被仿真 CPU 为 875X 时,此信号为内部 PSEN
WR: CPU 的写信号
S1:指令的第一个机器码,表示一个新的指令开始执行.
ALE:CPU 的 ALE 信号
PSEN:CPU 的 PSEN 信号
USER:当此信号为高时表示为用户状态,为低时表示监控状态.

☆对于 **POD-80C552** 仿真头:

其它信号与 8X5X 相同,P2 表示 P2 口的数据.

☆对于 **POD-87C520P** 仿真头:

其它信号与 8X5X 相同,
D: 表示内部数据总线值.
NMIACK: NMI 响应信号, 为高时表示为用户状态,为低时表示监控状态
PSENA: PSEN 信号(此时不分 CPU 为 80C520 还是 87C520)

☆对于 **POD-80C196** 仿真头:

P3 组: P3 端口
P4 组: P4 端口
WR: 写信号
BHE: BHE 信号
ALE: ALE 信号
RD: 读信号
INST: INST 信号
NMI: NMI 信号
USER: 程序状态,为高时表示为用户状态,为低时表示监控状态

☆对于 **POD-LPC76X** 仿真头:

AD 组: 地址/数据总线(功能与 8051 的 P0 端口一样)
EMAH 组: 地址高位总线
EMRD: 仿真器读信号(PSEN 信号与仿真芯片的 RD 信号相同)
EMWR: 仿真器写信号
ALE: ALE 信号
USER: 程序状态,为高时表示为用户状态,为低时表示监控状态

☆对于 **POD-PIC** 系列仿真头:

ADH: 地址/数据总线高字节
ADL: 地址/数据总线低字节
ALE: ALE 信号

PSEN: PSEN 信号

HACK: HALT 响应信号,高:用户状态,低: 监控状态

FNOP: 指令已经读入,但没有被执行.PIC 在执行当前指令时,就读入下一条指令等待执行,由于有跳转指令,下一条指令不一定会执行,所以会有指令读入而没有执行的情况

HALT: 程序停止执行

USER: 程序状态,为高时表示为用户状态,为低时表示监控状态

M0 标尺: 参考点标尺,可以将 M0 标尺设置在波形的指定点,通过移动 M1 标尺来测量波形的宽度,相对时间等.可以在波形图中拖动 M0 标尺来设置位置,也可用弹出菜单中的［设置 M0］功能来精确设置 M0 位置.在逻辑分析仪窗口的右下方,可以看到标尺在全局中的相对位置,红色竖线表示 M0 在全局波形中相对位置.M0 位置的数值(以时间为单位),M0 与 M1 的差值(以时间为单位),也可在窗口下方看到.击右键在弹出菜单中选择［转到/M0］功能,可将当前窗口可视区移到以 M0 为开始的位置.如果 M0 标尺在当前窗口可视区内,拖动 M0 标尺上方的方框,就可以在当前窗口可视区内移动 M0 标尺.(注:M0 标尺值是个固定值,不随着当前窗口可视区的移动而移动,当窗口移动时,M0 标尺可能会在可视区外)

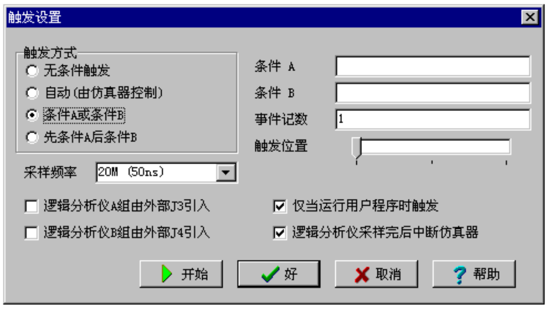
M1 标尺:相对值标尺,用于测量当前窗口可视区内某点位置,用 M1 标尺可以测量相对于 M0 标尺的时间差.M1 会随着当前窗口可视区移动而移动,在窗口右下方的全局位置区域内,绿色竖线表 M1 在全局波形中的相对位置.M1 标尺的时间值(以时间为单位)可在窗口下方看到. 拖动 M1 标尺上方的方框,就可以在当前窗口可视区内移动 M1 标尺.当称动 M1 标尺时,可以在信号名区域看到所在信号的状态,如果信号以组的形式显示,则信号组状态以十六进制值表示,如果信号以展开方式显示,则以 0 表示低平,1 表示高电平.

TRG 标尺: 触发点标尺,表示满足条件的触发点在全局中的位置,可以击右键在弹出菜单中选择［转到/触发点］功能,将当前窗口可视区移到以 TRG 为开始的位置.TRG 标尺是不可移动的.TRG 标尺的位置由所设置的条件决定的.

信号名: 此区域用于表示信号名称和信号值,如果移动 M1 标尺,显示的信号值为信号在 M1 标尺位置上的值,如果将鼠标移到信号名区域,则显示的信号值为当前实时采样到的值.信号值的显示方式根据总线是否展开也有所不同,如果是总线方式信号,则用十六进制值表示,如果是展开的单个信号,用″0″表示 M1 标尺位置上,信号电平为低,″1″表示 M1 标尺位置处,信号电平为高,在信号名上双击,可以展开/收缩信号的显示方式,功能与弹出菜单的［展开/收缩］相同.可以拖动信号名右边波形边框,扩大/缩小信号名

显示区域大小.

触发设置:



对逻辑分析仪工作方式进行设置.

[触发方式]:

1. 无条件触发:由用户手动控制逻辑分析仪启动/停止,当"开始"按钮被按下时,逻辑分析仪开始采样.当"停止"按钮被按下时或采样满 32K 后,逻辑分析仪停止采样.
2. 自动(由仿真器控制):逻辑分析仪的启动/停止由仿真器控制,也就是说逻辑分析仪随着用户调试的程序运行,逻辑分析仪才运行,程序停止,逻辑分析仪就停止.
3. 条件 A 或条件 B: 指定触发条件,可以使用所采样各路信号状态来组合触发条件.当触发事件条件中需要有'或'运算时,可加入条件 B,与条件 A 进行'或'运算使得事件定义更加灵活方便.(条件定义参照条件 A 说明)
4. 先条件 A 后条件 B:如果触发条件在时序上有先后要求时,就可以用这种方式.例: 在判断上升沿时,可以指定条件 A 为低电平条件,条件 B 为高电平条件.这样,先低后高条件即为上升沿. (条件定义参照条件 A 说明)

[**条件 A**]:定义事件触发条件. 对逻辑分析仪采样到的 40 路信号进行分析,如果某路信号或是某组信号出现了我们指定的状态,我们称之为条件, 例如,WR 信号由高变低,或者是 P0 口的值为 055H,都可以作为条件,也就是说所采样的 40 路信号的任意状态都可以做为条件来控制逻辑分析仪和仿真器. 更可以用条件 AB 的组合, 对复杂的状态进行分析. 对条件的定义可以用"非"(!)和"与"(&)进行组合.数字后加 H 表示十六进制,数字后加 B 表示二进制,二进制数字只能为 0 或 1.数字中的 X 为任意值.二进制数中的 X 表示相应位可以为 0 也可以为 1,十六进制数字中的 X 可以是 0 到 F 中的任意值.

注意:如果 X 位于数字的首位,请在 X 前加'0',否则会提示数据格式有错.

(例)

☆外接 J3 的第 0 位高, 而且 WR 信号为低的条件, 表示为:

J3.0 & !WR 或者 J3=0XXXXXXX1B & !WR

☆外接 J3 总线的值为 1AH, 而且 WR 信号为高的条件, 表示为:

J3=1AH & WR 或者 J3=00011010B & WR

☆外接 J3 低四位为二进制 1001,且外接 J4 的第 7 位为低, 表示为:

J3=1001B & !J4.7 或者 J3=0X9H & J4=0XXXXXXXB

☆外接 J3 高四位为二进制 0011,低四位任意值, 表示为:

J3=0011XXXXB 或者 J3=3XH 或者 !J3.7 & !J3.6 & J3.5 & J3.4

[**条件 B**]: 与条件 A 组合,对触发条件进行更加灵活的定义,按照触发方式不同,条件 B 可以与条件 A 进行'或'运算,定义事件的'或'关系,也可以定义事件时序的先后关系.从而定义信号的上升沿/下降沿.参见触发方式.

以 51 系列 CPU 为例,举例说明条件的定义

☆观察信号下降沿

观察在写信号下降沿时,程序运行状态,就要指定 WR 先高,后低。指定［触发方式］为"**先条件 A 后条件 B**",［条件 A］为"**WR**",意为 WR 为高,[条件 B]为"**!WR**",意为 WR 为低,这样就组成了 WR 的下降沿. 同时选中［**仅当运行用户程序时触发**］和［**逻辑分析仪采样完后中断仿真器**]两个条件,表明程序运行时,才判断是否满足触发条件, 而且当逻辑分析仪采样完后,程序也停止下来.［触发位置］设在**最左边**.表示有 WR 下降沿后才采样波形.以观察 WR 下降后,程序运行情况. 设置好后按［**开始**］键,仿真器等待条件出现,这时运行程序,当程序运行到指令 MOVX @DPTR,A 时,出现 WR 信号下降沿,条件满足,这时逻辑分析仪开始采样数据.采样完后中断程序,将采样数据上传到计算机,在逻辑分析仪窗口中就可以看到,WR 下降沿后,程序运行状态波形.

☆设置多次断点

在地址为 1234H 处停止下来,指定［触发方式］为**先条件 A 后条件 B**, ［条件 A］定义为 **AH=12H & AL=34H**,［条件 B］为**空**.［事件记数］为 **3**, 指明当程序第三次执行到 1234H,才满足条件. 同时选中［**仅当运行用户程序时触发**］和［**逻辑分析仪采样完后中断仿真器**］两个条件,表明程序运行时,才判断是否满足触发条件,而且当逻辑分析仪采样完后,程序也停止.［触发位置］设在**最右边**.表示先采样波形,以观察在第三次程序运行到 1234H 之前, 线路上的逻辑状态.设置好后按［**开始**］键,仿真器等待条件出现,这时开始运行程序,当程序运行时,也采样数据,当第三次到达 1234H 地址时,程序停止,然后上传采样到的数据.在逻辑分析仪窗口可以看到第三次运行到 1234H 之 前,程序运行情况.(注意:程序中 1234H 一定要是有效的指令地址,不能是一条多字节指令的中间地址).

[**事件记数**]:指明事件发生的次数,即在指定条件发生指定次数后,才触发事件.对信号进行采样控制.在程序运行时,同样的条件可能会出现多次,用这种方法可以指定触发条件出现多少次后再采样,从而可以方便准确地捕捉程序中的错误.

(例): 一般在程序中设置断点,只要程序一经过断点就会停止,可以用事件记数功能实现断点的多次停止功能,也就是指定程序经过断点多次后才停止程序而不是一次就停止.方法是将条件定义为程序地址, AL=地址低位 & AH=地址高位. 在事件记数中指定几次后程序停止,同时选中[**仅当运行用户程序时触发**]和[**逻辑分析仪采样完后中断仿真器**]两项. 并且[**触发位置放**]在右边.这样就会在程序经过断点多次后,才停在断

点处.

触发位置: 确定事件发生与信号采样的关系,分三种情况,

1. 当触发点在左边时,表示在触发条件满足后,才采样 32K 深度.所采样的数据为触发条件之后,程序运行产生的数据.
2. 当触发点在中间时,表示触发条件满足前采样 16K 深度,触发条件满足后再采样 16K 深度.
3. 当触发点在右边时,表示在触发条件满足前采样 32K 深度.所采样的数据为触发条件之前,程序运行产生的数据.

采样频率:选择逻辑分析仪的采样频率. 共有(20M,10M,1M,100K,10K,1K,J3.0,!J3.0)八种频率可选,可以根据 CPU 工作时钟做适当选择.也可以通过 J3.0 从外部引入采样时钟,J3.0 表示上升沿控制,!J3.0 表示下降沿控制.

逻辑分析仪 A 组由外部 J3 引入: 用户如果想采样外部工作信号,请将探钩接到 J3 插口,选中此项,就可以用逻辑分析仪采样分析外接用户信号,采样的到信号显示在 A 组(第一组)位置上,组信号名为 J3,各信号颜色与外接探钩线颜色相同.组信号名和组内各信号名称可以由用户自行定义. (外接逻辑探勾为选配件)

逻辑分析仪 B 组由外部 J4 引入: 用户如果想采样外部工作信号,请将探钩接到 J4 插口,选中此项,就可以用逻辑分析仪采样分析外接用户信号,采样的到信号显示在 B 组(第二组)位置上,组信号名为 J4,各信号颜色与外接探钩线颜色相同.组信号名和组内各信号名称可以由用户自行定义.

仅当运行用户程序时触发:由于逻辑分析仪可以独立于仿真器工作,所以如果想在用户程序停止运行时,采的"停止/开始"按键控制. 一般情况下,应选中此项,以免在用户程序停止时,误采信号.

逻辑分析仪采样完后中断仿真器:在逻辑分析仪满足事件条件,并且采样信号结束后,立即停止仿真器,上传逻辑分析仪数据,进行分析.如果没有选中此项,则在逻辑分析仪满足事件条件后，采样信号结束，逻辑分析仪将采样到的数据立即上传，而勿需等到程序执行终止. 一般情况下,应选中此项,这样逻辑分析仪的波形和断点处的程序有一定的关系,可以更加准确地查到错误,使分析更加方便.

停/开始:开始/停止逻辑分析仪进行采样,由于逻辑分析仪可以独立地进行工作,所以它不一定是靠仿真器来控制启动/停止采样,可以用逻辑分析仪窗口上的按钮来控制. 如果在逻辑分析仪触发设置中,没有选中"仅当运行用户程序时触发"或使用"无条件触发",就要通过这个键来控制逻辑分析仪的采样.

放大/缩小:对采样到的波形放大/缩小.当想观察波形局部细节,详细地分析信号逻辑状态,了解信号时序,计算脉冲间距,就要对波形放大(ZOOM IN),如果相了解程序运行的概况,观察波形全局情况,或者想对 MO 标尺做大致定位,就要对波形缩小(ZOOM OUT).

弹出菜单

设置 M0：设置 M0 标尺,在检测波形时,可以以 M0 作为参考点,而以 M1 作为相对标尺来测量波形的宽度.(设置对话框)

转到： 将当前波形显示窗口,移到以 M0 或以 TRG(触发点)为开始的地方,方便迅速定位窗口,查看波形.

展开/收缩:展开/收缩当前位置的波形.当波形收缩显示,即以组方式显示时,可以看到总线上八根线的状态所组成的值,(例)对于 AL 组(地址总线低八位),值为 0ABH,那么对于 AL.7 到 AL.0 的各条线的的高低电平为 1010 1011. 当波形展开时,以单路信号线显示,可以直接看到总线上每条线的高低电平,对于控制线的分析更加直观有效.也可以双击左边的信号名,展开/收缩相应信号. 修改信号名：用户可以修改外接的信号名,适应用户不同的应用电路.为分析信号提供更加方便快捷的手段.

查找条件：查找 40 路信号中,满足指定条件的位置,用来检查这些条件是否出现过,或条件出现前后各路信号的状态是什么,用这种方法可以迅速定位所想要查找的状态。对条件的定义可以用"非"(!)和"与"(&)进行组合.数字后加 H 表示十六进制,数字后加 B 表示二进制,二进制数字只能为 0 或 1.数字中的 X 为任意值,二进制数中的 X 表示相应位可以为 0 也可以为 1,十六进制数字中的 X 可以为 0 到 F 中的任意值.

注意:如果 X 位于数字的首位,请在 X 前加'0',否则会提示数据格式有错.

（例）

☆外接 J3 的第 0 位高， 而且 WR 信号为低的条件， 表示为:

J3.0 & !WR 或者 J3=0XXXXXXX1B & !WR

☆外接 J3 总线的值为 1AH， 而且 WR 信号为高的条件， 表示为:

J3=1AH & WR 或者 J3=00011010B & WR

☆外接 J3 低四位为二进制 1001,且外接 J4 的第 7 位为低， 表示为:

J3=1001B & !J4.7 或者 J3=0X9H & J4=0XXXXXXXB

☆外接 J3 高四位为二进制 0011,低四位任意值， 表示为:

J3=0011XXXXB 或者 J3=3XH

查找下一个条件：查找指定条件下一次出现的位置.

触发设置：对逻辑分析仪进行设置.(设置对话框)

备份 ：将某一组信号备份到 F0 组,做为参考,也可以与下次采样到的波形进行比较。F0 组数据不随着采样而改变，只有在重新备份时,F0 组的数据才会改变.

调入波形文件 ：装入保存的波形等进行分析.

保存波形文件 ：对采样到的波形进行保存,以备下次分析作参考.

打印波形 ：打印采样到的波形.

逻辑探勾（选配件）

逻辑探勾用于采样外部逻辑信号,使用时,将探勾插在 J3 或 J4 处,同时请在触发设置窗口里,选中[逻辑分析仪 A 组由外部 J3 引入]或[逻辑分析仪 B 组由外部 J4 引入] (参见“触发设置”)(注:E51/L 只能外接一组 J3),这样共可以外接 16 路信号进行采样.当使用外接探勾时,在逻辑分析仪窗口里,A 组或 B 组的信号名变成 J3 或 J4,而不是原来内部信号名.对于展开后的各信号名,可以根据设计需要自己定义,信号波形的颜色与探勾上线的颜色相同,依次为**红橙黄绿蓝紫灰白**,分别对应外接信号组的第 **01234567** 位,探勾线上还有两根未接探头的空线,一根褐色为+5V,一根黑色为地.一般情况不需要接这两根线.

应用举例:

☆用自动方式测量端口计数波形

1. 输入以下程序,并存盘.这个程序将对 P1 口计数.

```
        ORG   0000H
        MOV   P1,#0
        MOV   B,  #100
LOOP:   INC   P1
        DJNZ B,  LOOP
HERE:   JMP   HERE
        END
```

2. 新建项目,将程序作为模块加入到文件中,设置仿真头类型,CPU 类型,保存项目
3. 选择菜单"仿真器 | 跟踪器/逻辑分析仪设置",将工作方式设为[逻辑分析仪],
4. 打开逻辑分析仪窗口,选择逻辑分析设置功能,将[触发方式]设为**自动(由仿真器控制)**同时选中[**逻辑分析仪 A 组由外部 J3 引入**], [**仅当运行用户程序时触发**]和[**逻辑分析仪采样完后中断仿真器**]三个条件.[采样频率]**选择 10MHz(100ns)"**
5. 将外接逻辑探头接在 J3 插口,逻辑探勾 0 号探针(红色)接仿真头 P1.0,将 1 号(橙色)探针接仿真头 P1.1,依次将其它探针接到 P1 口其它管脚.
6. 按 F9 键,编译程序.光标移到"HERE"标号处,按 Ctrl-F8 设置断点.
7. 全速运行程序,程序在"HERE"处停止.
8. 打开逻辑分析仪窗口,就可以看见第一组(J3)的值从 0 到 100 递加.

☆ 通过下面的程序, 我们用逻辑分析仪观察一个写周期及一个串行口数据的发送. 为了叙述方便,我们在前几条语句后加上了该条指令的地址及机器码.

```
        ORG       0
        MOV       DPTR, #1234h        ; 0000   90 12 34
        MOV       A,  #56h            ; 0003   74 56
        MOVX      @DPTR,  A           ; 0005   F0
        NOP                           ; 在这行设第一个断点
        MOV       TMOD,#020H          ; TIMER1: 方式 2 (8 位重载)
        MOV       TH1,#0F3H           ; 2400BPS @ 12MHz
        MOV       TL1,#0F3H
        ANL       PCON,#07FH          ; 清 SMOD 位
        MOV       SCON,#50H
        SETB      TR1

        MOV       A,  #055H
        MOV       SBUF,  A            ; 发送 55H
```

```
Wait:   JNB     TI, Wait
        SJMP    $               ; 在这行设第二个断点
        END
```

在第五行及第十七行设置两个断点

首先设置采样频率 20M，全速执行到第一个断点，这时逻辑分析仪采样到为前三条指令的执行波形，下页我们会结合波形详细说明．接下来我们来采样串行口发送出的数据波形，我们在程序中设置的波特率为 2400，每一个数据位的长度应为：

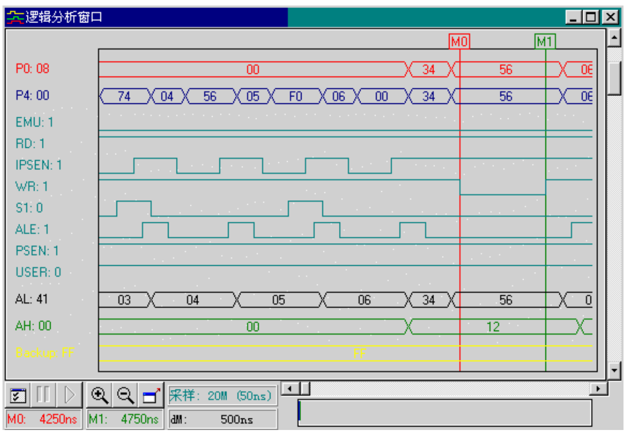
1000000us/2400 = 416.667us

串行口的发送数据的格式为一个起始位 0，8 个数据位(低位在前，高位在后)，一个停止位 1，共十位．我们在程序中发送了 55H，因此串行口数据为：0，1,0,1,0,1,0,1,0，1 我们设置采样频率为 1M，选择逻辑分析仪 A 组由外部 J3 引入，将 J3 第 0 位测试钩(红色) 钩在仿真头第 11 脚(TXD 引脚)上，全速执行到第二个断点．请看下页波形．

下图显示的是 MOVX @DPTR, A 的执行波形，这条指令的地址为 0005，机器码为 F0.

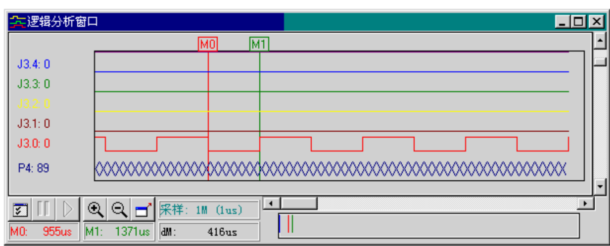
图中 AH，AL，P0 分别为高位地址，低位地址，P0 口.

1. 在 ALE 的上升沿后 P0 口输出地址 05，ALE 下降沿，地址被锁存.
2. 然后，PSEN 变低，开始读指令，稍后机器码 F0 被读出.
3. 接下来读入的 0006 单元 00 为 MCS51 CPU 取的下一条指令，被丢弃

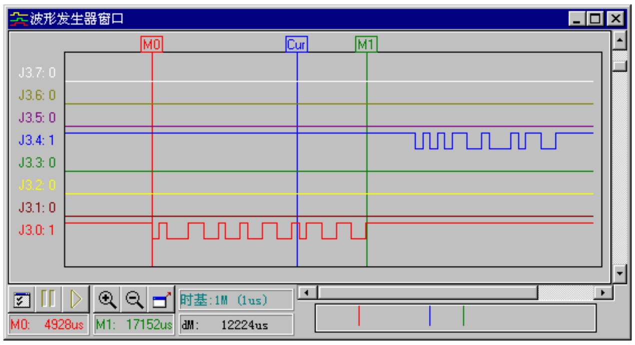


4. CPU 送出要访问单元地址 1234H，接着送出 A 的内容 56H，并输出写信号
5. 将 M0，M1 分别移至 WR 信号上升/下降沿．此时 dM：显示的为 WR 的宽度为 500ns

上图显示的是串行输出脚上的输出波形．可以看出输出的串口数据为：0,1,0,1,0,1,0,1,0,1 最后一个停止位 1 未输出完是因为MCS51 CPU在开始发送停止位后就置位 TI 的原故。移动 M0，M1 可测出一个数据位的宽度为 dM：416us，并与理论数据 416.667us 相吻合.



波形发生器



波形发生器是一种数据信号发生器,在调试硬件时,常常需要加入一些信号,以观察电路工作是否正常.用一般的信号发生器,不但笨重,而且只发一些简单的波形,不能满足需要.例如用户要调试串口通信程序时,就要在计算机上写好一段程序,再用线连接计算机和用户实验板,如果不正常,不知道是通讯线有问题还是程序有问题.用 E6000/L 的波形发生器功能,就可以定义串口数据.通过逻辑探勾输出,调试起来简单快捷.

将逻辑探勾接在 J3 插槽上,波形发生器通过 J3 可以输出 8 路自定义数字波形,每路可以单独加在用户板的任何**输入**端. 波形发生器可以选择不同的时间基数,做为定义波形的最短间。E6000/L 可产生最短时基为 50ns.可自定义波形长度为 时基 x 32767. 波形发生器可自动返回,环产生波形. 波形发生器窗口如上图

注意:当用探勾做波形发生输出时,一定要将探勾接在器件的输入端,
而不能接在输出端,否则会导致仿真器损坏.如果是双向器件,
请注意,波形发生器工作时,器件的数据走向.

M0 标尺:参考点标尺,可以将 M0 标尺设置在波形的指定点,通过移动 M1 标尺来测量波形的宽度,相对时间等.可以在波形图中拖动 M0 标尺来设置位置,也可用弹出菜单中的[设置 M0]功能来精确设置 M0 位置.在逻辑分析仪窗口的右下方,可以看到标尺在全局中的相对位置,红色竖线表示 M0 在全局波形中相对位置.M0 位置的数值(以时间为单位),M0 与 M1 的差值(以时间为单位),也可在窗口下方看到.击右键在弹出菜单中选择 [转到/M0]功能,可将当前窗口可视区移到以 M0 为开始的位置.如果 M0 标尺在当前窗口可视区内,拖动 M0 标尺上方的方框,就可以在当前窗口可视区内移动 M0 标尺. (注:M0 标尺值是个固定值,不随着当前窗口可视区的移动而移动,当窗口移动时,M0 标尺可能会在可视区外)

M1 标尺:相对值标尺,用于测量当前窗口可视区内某点位置,用 M1 标尺可以测量相对于 M0 标尺的时间差.M1 会随着当前窗口可视区移动而移动,在窗口右下方的全局位置区域内,绿色竖线表 M1 在全局波形中的相对位置.M1 标尺的时间值(以时间为单位)可在窗口下方看到.拖动 M1 标尺上方的方框,就可以在当前窗口可视区内移动 M1 标尺. 当称动 M1 标尺时,可以在信号名区域看到所在信号的状态,如果信号以组的形式显示,则信号组状态以十六进制值表示,如果信号以展开方式显示,则以 0 表示低平,1 表示高电平

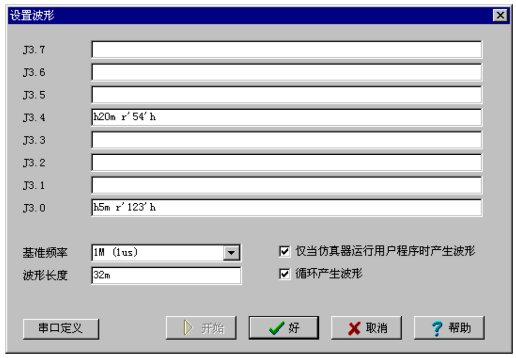
Cur 标尺:当前点标尺,表示正在发送波形的位置, 程序运行时,Cur 标尺会随着程序运行而向右移动. 如果设置为[循环产生波形],那么标尺移动到最右边后,就会自动重新从头开始产生波形,可以在弹出菜单中选择[设置 Cur]功能,来设置 Cur 标尺位置,这样就可以重新指定波形开始点,当程序开始时,波形就从指定点产生,使调试更加方便.可以击右键在弹出菜单中选择[转到/当前点]功能,将当前窗口可视区移到以 Cur 为开始的位置.当程序运行时,Cur 标尺会指出当前波形位置,这样用户可以清楚看到,波形发生器所发出的波形.

J3 信号名:此区域用于显示 J3 信号名称和信号值,如果是总线方式信号,则用十六进制值表示,如果是展开的单个信号,用"0"表示 M1 标尺位置上,信号电平为低,"1"表示 M1 标尺位置处,信号电平为高,在信号名上双击,可以展开/收缩信号的显示方式,功能与弹出菜单的[展开/收缩]相同.可以拖动信号名右边波形边框,扩大/缩小信号名显示区域大小.

波形发生器控制字使用方法:

控制字	表达意义	例	说明
H	高电平	h100n	高电平保持 100ns
		h	保持高电平到结束
L	低电平	L1m	低电平保持 1ms
S	从某时间开始	s100u	以下定义从 100u 开始
R	RS232 数据	r0	仿串行口发出数据 0
		r'abc'	仿串行口发出数据'a','b','c'
(	重复开始符号	(H10u L10u)10	重复 10us 高,10us 低,共十次
		(H10u L10u)	重复 10us 高,10us 低,直到结束
)	重复结束符号		
{	段开始符号	{1m, 3m}	以下定义只适用于 1ms 到 3ms 时间段
		{1m, 3m}(H10uL10u)	在 1ms 到 3ms 时间内为高 10us 低 10us 脉冲
}	段结束符号		
n	纳秒(ns)		
u	微秒(us)		
m	毫少(ms)		

波形定义对话框



[**基准频率**]:

最小时间单位,在这个时基下,波形所能产生最短时间宽度.

[**波形长度**]:

能自定义波形的最大长度, 最大长度必需小于 32767 * 时基,例如当基准频率为 1us 时,最大可产生的波形长度为 32.767ms.

[**仅当运行用户程序时产生波形**]:

如果选中此项,波形发生器只有在用户程序运行时,才产生波形,如果去掉选择,波形发生器一直工作,这时由[开始]和[结束]按键控制.如果此时想一直输出波形,请选择[循环产生波形],



[**循环产生波形**]

如果选择此项,波形发生器在到达波形长度后,会自动从头开始产生波形.。如果没有选择此项目,波形发生器将在到达波形长度后就停止.

[**串口定义**]

当使用 R 命令产生串口波形时,用来定义串口协议.

波形定义举例:

1. H100n L200n H300n L400n H
   高 100ns, 低 200ns, 高 300n, 低 400n, 保持高直到结束
2. H10m (H100u L200u)2 (H200u L500u)
   高 10ms, 重复(高 100u, 低 100u)2 次, 重复(高 200us, 低 500us)直到结束
3. (h100u (h200u l300u)10)3 h
   重复(高 100us 重复(高 200us, 低 300us)10 次) 3 次, 保持高直到结束
4. (h100u l100u) {1m, 3m} (h10u l10u)
   重复(高 100us,低 100us)直到结束,其中在 1ms 到 3ms 区间内为(高 10us,低 10us)脉冲.
5. h1m r0 r1 r'aB' r '012' h
   高 1ms,发串口数据 0,1,'a','B','0','1','2', 高电平直到结束.串口通信协议由[串口定义]设定
6. (h100n l100n) 时基设为 100ns,波形长度设为 200ns,同时选中[循环产生波形]
   用这种方法可以简单地循环产生一个高 100ns,低 100ns 的时钟方波。

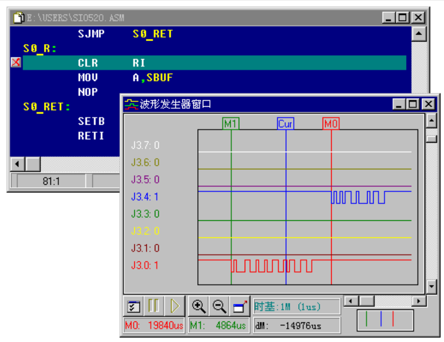
应用举例:

以 51 系列串口接收为例,用波形发生器产生一个 RS232 串口波形,用 CPU 的 RXD 口来接收这个串行码,来调试程序.

1. 输入以下程序,并存盘

```
         ORG      0000H
         LJMP     START
         ORG      0023H
         LJMP     SIO_0
START:   MOV      IE,#0          ;DISABLE ALL INTERRUPT
         MOV      TMOD,#020H   ;TIMER1 MODE2 (8BIT RELOAD)
         MOV      TH1,#0F3H    ;2400BPS @ 12MHz
         MOV      TL1,#0F3H
         ANL      PCON,#07FH   ;CLEAR SMOD BIT FOR SIO0
         MOV      SCON,#50H
         SETB     TR1
         SETB     ES
         SETB     EA
         SJMP     $

SIO_0:   CLR      RI
         MOV      A,SBUF
         NOP
         SETB     ES
         RETI
         END
```



2. 新建项目,将程序作为模块加入到文件中,仿真头类型,CPU 类型,保存项目

3.编译,查出拼写错误.

4.选择菜单 ”仿真器 | 跟踪器/逻辑分析仪设置”,将工作方式设为**[波形发生器]**

5.打开波形发生器窗口(与逻辑分析仪窗口的图标是相同的).打开设置对话框.

按如下设置:

[**基准频率**]:设为 1MHz(1us)　　[**波形长度**]:设为 32ms

[**仅当仿真器运行用户程序时产生波形**] 选中　　[**循环产生波形**] 选中

打开[**串口定义**]对话框,[**波特率**]选择 2400BPS, [**数据位**]8 个, [**校验位**]无, [**停止位**]1 个.

在[J3.0]信号栏内填写 **”h5m r’123’ h”**,意为先保持 5 毫秒高电平,主要用来等待程序初始化,然后送出串行信号,数据为 ASCII 字符 1,2,3,再保持高电平

6. 将逻辑探勾插在 J3 处,并将红色探勾接在 CPU 的第十脚(RXD)
7. 在串口中断程序内 NOP 指令处,设置断点.光标移到串口中断程序内 NOP 行,按 Ctrl-F8
8. 按 F9 编译,按 Ctrl-F9 全速运行程序.
9. 程序在中断程序 NOP 指令处停止,观察 A 寄存器值,

6 DOS 版软件使用

一 集成调试软件使用说明

1.1 安装盘内容

安装盘上与DOS软件有关的文件夹如下:

文件夹	文件内容
A51	MCS51 系列汇编调试软件
A96	MCS96 系列汇编调试软件
P51	MCS51 系列高级语言调试软件. 可以调试 ASM, PL/M, C 语言
P96	MCS96 系列高级语言调试软件. 可以调试 ASM, PL/M, C 语言
LPC76X	旧式LPC仿真/编程器软件
PIC	PIC 系列汇编调试软件

1.2 软件安装

1.2.1安装 MCS51 系列汇编调试软件

键入	执行结果
C:	进入 C 盘. 这里设要将调试软件安装至 C 盘
MD MCS51	建立 MCS51 子目录
CD MCS51	进入 MCS51 子目录
将光盘相应目录下内容复制到C：盘	执行ATTRIB -R *.* 将目录所有文件的只读属性去掉。

可以在光盘自动启动后，选择“安装DOS软件”功能，在安装DOS窗口中选择要安装的软件，指定安装目录，确认后，计算机会自动将DOS软件安装到指定位置。

6

1.2.2 安装 MCS96 系列汇编调试软件

键入	执行结果
C:	进入 C 盘. 这里设要将调试软件安装至 C 盘
MD MCS96	建立 MCS96 子目录
CD MCS96	进入 MCS96 子目录
将安装盘插入 A 驱 A:A96 A:TEST	执行 A 盘上的 A96 文件, 将汇编调试软件安装至 MCS96 子目录中. 如果使用 B 盘安装请键入 B:A96

1.2.3 安装 MCS51 系列高级语言调试软件

键入	执行结果
C:	进入 C 盘. 这里设要将调试软件安装至 C 盘
MD 51	建立 51 子目录
CD 51	进入 51 子目录
将安装盘插入 A 驱 A:P51 A:TEST	执行 A 盘上的 P51 文件, 将高级语言调试软件安装至 51 子目录中. 如果使用 B 盘安装请键入 B:P51

1.2.4 安装 MCS96 系列高级语言调试软件

键入	执行结果
C:	进入 C 盘. 这里设要将调试软件安装至 C 盘
MD 96	建立 96 子目录
CD 96	进入 96 子目录
将安装盘插入 A 驱 A:P96 A:TEST	执行 A 盘上的 P96 文件, 将汇编汇调试软件安装至 96 子目录中. 如果使用 B 盘安装请键入 B:P96

1.2.5 安装 PIC 系列 汇编语言调试软件

键入	执行结果
C:	进入 C 盘. 这里设要将调试软件安装至 C 盘
MD PIC16	建立 PIC16 子目录
CD PIC16	进入 PIC16 子目录
将安装盘插入 A 驱 A:PIC	执行 A 盘上的 PIC 文件, 将高级语言调试软件安装至 PIC16 子目录中. 如果使用 B 盘安装请键入 B:PIC

1.3 集成调试软件介绍

对于不同的单片机,集成调试软件会有些不同. 不同之处主要在 CPU 窗口, 因为不同的单片机的内部结构有所不同. 另外对于汇编语言与高级语言,集成调试软件也存在差别. 汇编语言调试软件与高级语言的主要差别在于它们对符号表处理的不同上, 因此汇编语言调试软件与高级语言调试软件的观察窗口(WATCH)是不同的. 下面我们以 MCS51 系列汇编调试软件为例来编写本手册. 不同之处再着重说明.

图2.3.1 为屏幕显示的MCS51系列集成调试软件，它包含三部分：

1. 顶部的一行为菜单栏
2. 中间的部分为工作窗口区
3. 底部的一行为常用命令快捷键提示栏

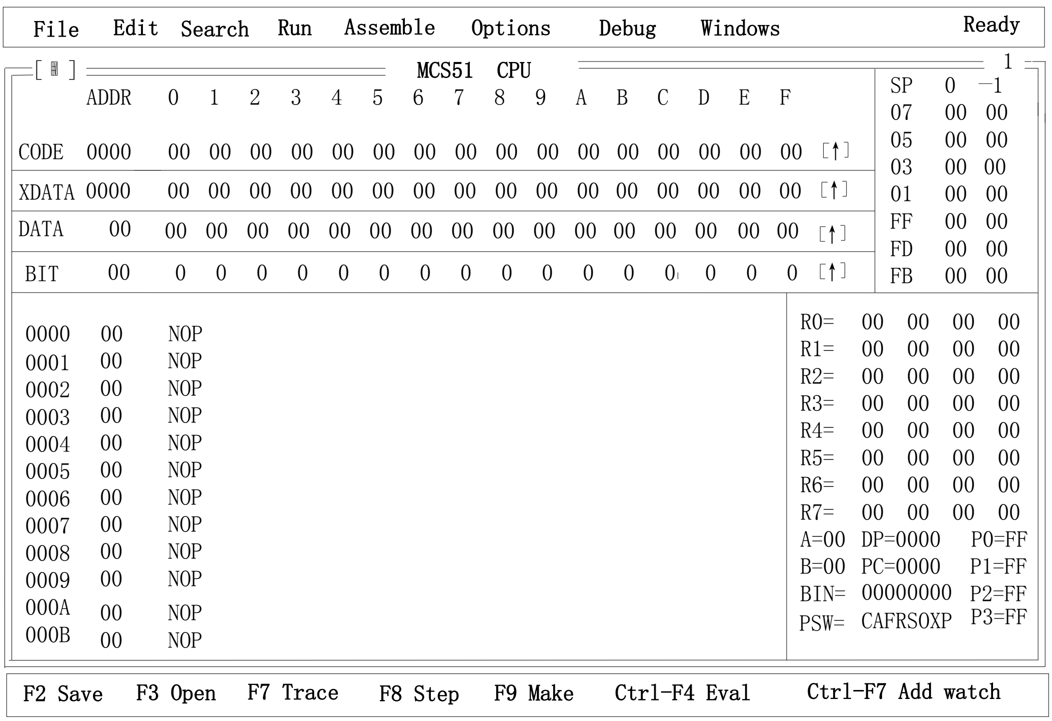


图 2.3.1 MCS51系列的集成环境

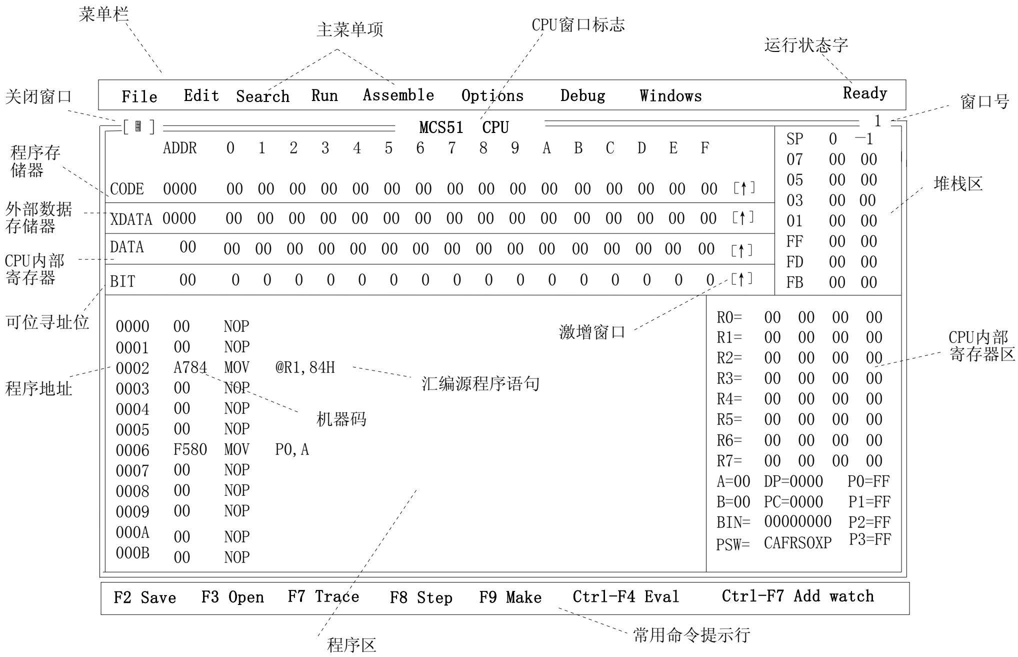


图 2.3.2 软件集成环境说明

下面结合图 2.3.2，分别介绍集成环境各个部分的作用。

1.3.1 菜单栏

菜单栏中含有主菜单项和运行状态字两部分内容:

1.3.1.1 主菜单项

主菜单项占据了菜单栏的大部分,File、Edit…Window等均为主菜单项,主菜单是软件的最上层菜单。

1.3.1.2 运行状态字

运行状态字位于菜单栏的最右端,状态字根据软件运行状态给出相应的提示信息.

Ready	表明可以进行操作	LA	表明逻辑分析仪正在工作
Wait	表明请稍候	LA RUN	表明仿真器与逻辑分析仪在同时工作
PM	表示在下装用户程序	RL	表明逻辑分析仪正在上装波形
Running	表明仿真器正在运行用户的程序		

1.3.1.3 菜单命令的执行方式

菜单栏提供编辑和调试软件时要用到的各种命令,命令是以菜单方式给出的.当菜单栏中的某个菜单被选中后,将弹出下层菜单,如图2.3.2为选中菜单栏中Search项后弹出的下层菜单.有些下层菜单还有更下层的菜单.第二层菜单中有两类命令:直接命令和对话框命令.

直接命令方式

菜单中各条命令项后不带有省略号"…"的称直接命令项,如图2.3.3"Search again"命令，选中它后便可直接执行它指定的命令.

对话框命令方式

在弹出的下层菜单中如果某项的后面有省略号"…",则表明选中该项后将弹出一个对话框,如图2.3.3中的"Find"则是带有对话框命令项.对话框的形式有多种,对话框的作用是要求用户对某些软件操作的要求作出选择、或者输入有关的信息,这些信息不能用简单的"是"或"不是"来表达,必须输入由用户决定的内容.例如,对话框要求用户输入的是文件名等信息.图2.3.4中间部分为Sesrch\Find命令的对话框。

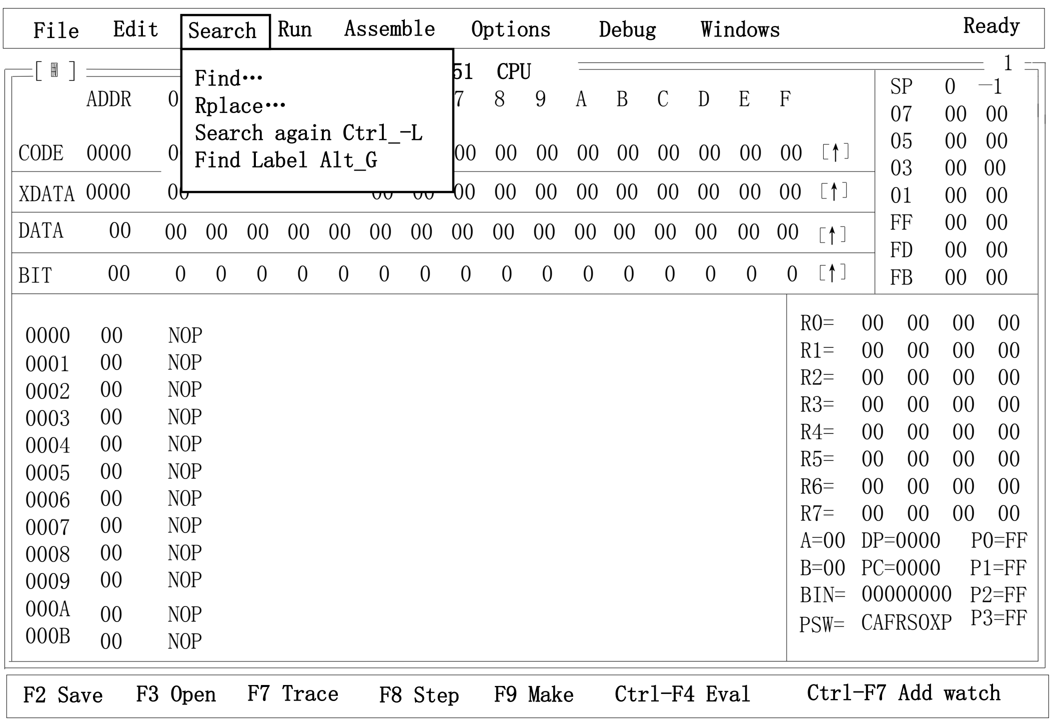


图2.3.3 Search 的下层菜单

1.3.2 对话框介绍

对话框中共有6种与用户对话的形式:

1.3.2.1 操作命令框

图2.3.4的中间部分的方框为选择Search\Find项后弹出的对话框，选择它后可执行查找字串的操作。其中写有OK和Cancel的两个方框为两个操作命令框,选择OK执行查找字串的命令,选择Cancel则放弃所有的选择,不执行任何命令返回.

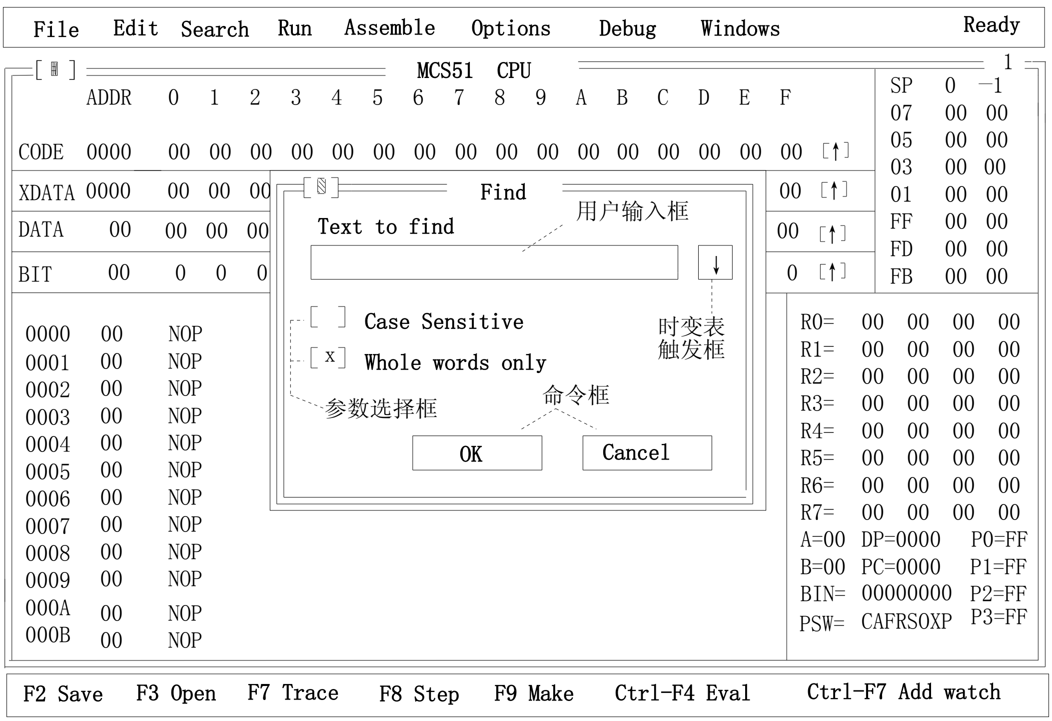


图2.3.4 Search\Find 菜单项的对话框

1.3.2.2 用户输入框

图2.3.4中的用户输入框由用户输入有关信息,例如,本对话框应由用户输入要查找的字符.

1.3.2.3 列表框

图2.4.5是File\Open项弹出的对话框，中间的部分为列表框，用户可从列表框中选择一个已经存在的源文件，选择源文件时先用鼠标在欲选的文件名上点一下，再用鼠标点命令操作框 “ Open ".其他菜单中弹出的列表框中也可能是其他的内容，如磁盘的子目录等.

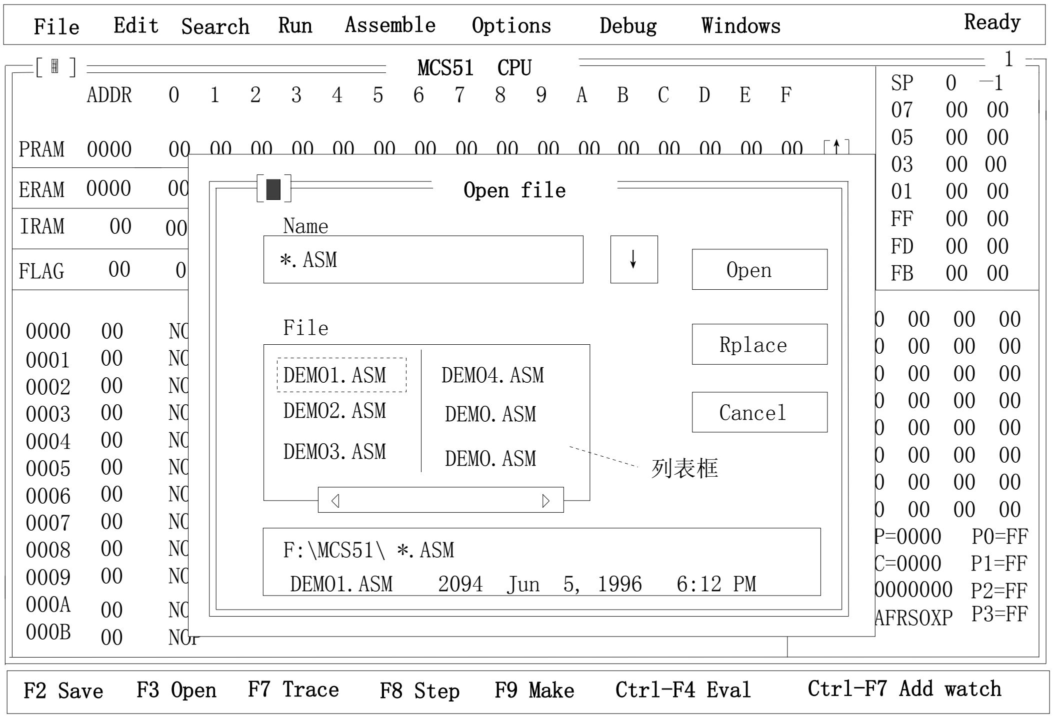


图2. 3. 5 对话框中的列表框

1. 3. 2. 4 命令参数选择框

图2.3.4中的命令参数选择框要求用户选择一项或多项命令选项,选择时用鼠标在“[ ]”号中点一下，选中后"[ ]"标记有" X "字符,未选择的为空白.解除选择时用鼠标在“[ ]”内再点一次即可.图中已选中"Whole words only"项.

1. 3. 2. 5 命令参数单选框

图2.3.6中标有“( )”的选项为命令参数单选框,在多个标有( )号的选项中只能选择其中的一项，各个选项之间是互斥的。图2.3.6选中的是 “ With address " .

1. 3. 2. 6 时变表触发框

图2.3.4和图2.3.5中标有 “ [↓] ” 的是时变表触发框,用鼠标选择它后,将用户输入窗口放大,并显示出在本次输入之前已经输入的信息,它实际上是一个记录表,记录了本次开机后用户输入的所有信息.

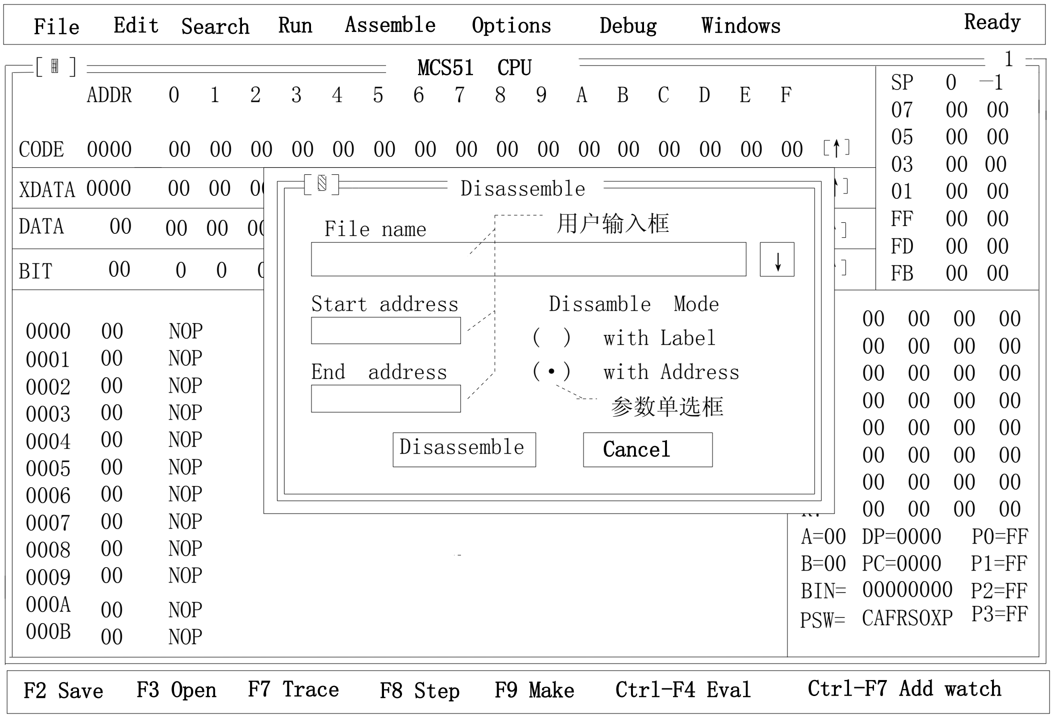


图2.3.6　对话框中的命令参数单选框

1.3.3 对话框的操作

如果菜单命令后省略号（…），选择它后将打开一个对话框，使用对话框可以方便地浏览和设置多个选择项。

1.3.3.1 命令框的操作

对话框中一般有二个标准动作按钮：OK、Cancel.如果选择OK，则确认了对话框中的各项选择；如果选择Cancel，各项原有选项的内容不变，也不执行任何命令，退出对话框。按ESC键与选Cancel是等效的。

使用键盘时，可按下某项中高亮度字符选择之。按TAB键可使光标在对话框中从某项移到另一项。使用鼠标时，在预选的命令框中按下鼠标左键即可。

1.3.3.2 用户输入框的操作

输入框是用户打入正文的地方。绝大多数基本的正文编辑键均适用于输入框。如果输入的信息超过框的长度，那么框中的内容自己滚动。如果输入框右边有一个下拉键头↓，那么将弹出一个记录用户已往输入信息的时变表，按下↓浏览时变表，双击鼠标按钮，可以从浏览表中选择一项用户前几次使用输入框时打入的正文，这条正文便自动调入用户输入框，可省去用户重复输入这条正文的操作，在时变表外单击鼠标按钮或按ESC从时变表中退出。

1.3.3.3 列表框的操作

列表框如图2.3.5所示，它的作用是列出已有的文件或其他有关选项，供用户直接从表中选择，选中的项直接被调入对话框，可省去用户输入的步骤。

1.3.3.4 命令参数选择框的操作

在对话框中选择某个命令参数选择框时，其中出现字符“X”，表示该命令参数选择框被打开；命令参数选择框为空表明它被未被选中。在命令参数选择框中按下鼠标按钮，使用键盘时，按TAB键直到该命令参数选择框被高亮度显示，然后按下空格键，或直接选择高亮度字符，从而打开或关闭一个命令参数选择框。

1.3.3.5 命令参数单选框的操作

命令参数单选框与命令参数选择框区别仅在于，前者总是以互斥选项组的形式出现，只能有一个选项被打开。选择命令参数单选框的操作与命令参数选择框的操作相仿。

1.3.4 窗口区

1.3.4.1 窗口介绍

集成环境中的中间部分是窗口区,它可包含一个或几个窗口.图2.3.1窗口中只有一个窗口,即CPU窗口。CPU窗口又由单线划分成几个部分.图2.4.1则有4个窗口:1号窗口为CPU窗口、2号窗口为观察窗口(Watch)，3号和4号为两个不同的源程序编辑窗口.CPU窗口的编号固定为1号，观察窗口固定为2号，源编程编辑窗口编号从3开始，可以打开多个，每个打开的源文件占用一个窗口，可打开的窗口数受计算机内存限制，内存大则可打开的窗口就多。

每个窗口的最上面一行中间部分的文字标明窗口名称,如图一中标有的“MCS51 CPU”，表明它是名为“MCS51 CPU "的窗口.窗口的右上角的数字为窗口编号。

窗口区中的所有窗口只有一个是激活的，激活的窗口的四周为双线且为高亮度线。激活的窗口的左上角标有[ ]的符号是关闭该窗口用的,当鼠标点在它上面时,关闭这个窗口.窗口右上角的[↑]符号为激增窗,用于放大或缩小该窗口.各个窗口激活后也可移动.

集成环境允许设置多个窗口（窗口数目由内存容量决定），但在任何时刻，只能有一个窗口处于活动状态，用户可以在设置的多个窗口之间切换.用户选择的任何菜单命令和输入的任何信息，只是针对被激活的活动窗口进行的。

各个窗口的作用和对窗口的操作方法,将在介绍相关的命令时予以解释.

1.3.4.2 MCS51 系列 CPU 窗口

CPU窗口如图2.3.1所示，它包括下列8个部分：

1. ADDR 0…F ［地址栏］

最上面一行标有"ADDR 0 1 …F "是地址名称栏，“ADDR”表示集中在一行中16个存储器(或寄存器)单元的第一个单元的地址，“0…F”表示它的下面对应着16个存储单元（或寄存器），“0”下面对应的2位16进制数，是第一个地址中的内容，“1”下面对应的是第二个地址单元中的内容…，“F”下面是第16个存储单元的内容。

“ADDR” 下面如果是4位地址值，则表明这个地址是程序存储器或数据存储器的，如果地址值是2位，则表明是单片机内部各类寄存器地址。地址值左侧的字串”CODE”、”XDATA”、”DATA”和”BIT”表明这一行的地址是程序存储器地址、数据存储器地址、单片机内部寄存器地址和标志寄存器中的各个标志位地址。

2. CODE ［程序存贮器］

它是为用户观察程序存贮器内部信息而设置的,共有64KB的地址空间,但是，在一般情况下只显示16个单元的信息。“ADDR”栏下的是以16进制表示的4位地址，地址后的16个数是这个地址及其随后的16个存储单元的内容。如果希望同时看到更多的信息，可用"↓"或"↑"键选择下一行的16个地址。若用鼠标，可拖动右边的滚动条查看下面的16个单元，或单击右侧的激增框，将窗口放大到满屏幕，同时查看更多的内容。

当用鼠标选中某个地址单元后，直接按数字键或A…F键，将弹出一个窗口，用户可为该单元输入新的数值。

如果地址值较大，用鼠标翻页比较麻烦，可先用光标选中CODE窗口，再按ATL-G键，则可弹出一地址窗口，用户输入地址值后，光标将自动定位在要求的地址单元上，然后再修改这一单元的内容。

3. XDATA ［数据存储器］

它是为用户观察外部数据存贮器的信息而设置的,共有64KB的地址空间。其操作方法与程序窗口"CODE"相同。在仿真87/89系列单片机且P0，P2口作I/O用时，请关闭XDATA窗口。

当用鼠标选中某个地址单元后，直接按数字键或A…F键，将弹出一个窗口，用户可为该单元输入新的数值。

如果地址值较大，用鼠标翻页比较费时，可先用光标选中XDATA窗口，再按ATL-G键，则可弹出一地址窗口，用户输入地址值后，光标将自动定位在要求的地址单元上，然后再修改这一单元的内容。

4. DATA ［单片机内部数据存储器］

它包括单片机内部数据存储器共256字节,其中通用寄存器32字节(即4组R0～R7),地址为00H～1FH；直接可寻址位16个字节，地址为20H—2FH，专用寄存器（SFR）共128字节，地址为80H～FFH。其操作方法与程序窗口“CODE”相同。

5. BIT ［位寻址窗口］

8031单片机内部RAM块中的16个字节（RAM地址为20H—2FH）。其操作方法与［CODE］相仿，所不同的是，这时“0—F”栏下对应的是16位，而不是16个字节。

应注意的是，在对内部RAM进行位操作时（指查看和修改），位的地址值与内部RAM字节的地址的值是有区别的，例如，内部RAM中地址为20H的字节，若要位寻址，则位地址为00H，而不再是20H。

6. SP ［堆栈指针］

CPU窗口右上角是堆栈指针,最上面一栏的“SP”、"0"、"-1"三项,它们下面对应的分别是:①. "SP"下面对应的是堆栈地址；②. "0"下面的是堆栈地址中的内容；③. “-1”表示上一个堆栈地址，即"SP"下面的地址减1，"-1"下面对应的是上一个堆栈中的内容。例如，“SP”下面的地址是07“0”下面第一行的“00”是07H单元中的内容，“-1”下面的00是06地址中的内容。

进行堆栈操作时,“SP”下的第一行总是栈顶的地址,其下的各个地址也随之变化。若系统复位，则SP地址初始化为07H，考虑到08H～1FH属于工作寄存器区1～3，若程序设计中用到这些区，最好把SP值设为1FH或更大值。

7. 工作寄存器区

CPU窗口的右下角是寄存器区，标有“R0=”…“R7=”、“A”“B”等字样。它们可直接反映单片机中常用寄存器状态。这些寄存器内部的信息也可在“DATA”窗口中查到，但是，在这个窗口列出更为直观方便。它列出了41个寄存器的内容，分述如下：

①.”R0=”…“R7=”　通用寄存器
由于单片机内有4个工作区，每个区中都有R0…R7共8个寄存器，“R0=”…“R7=”右侧对应的是寄存器的内容，共分4列，每一列对应一个工作区中的8个寄存器中的数据。

②."A=　"　累加器
它表示累加器中的内容。

③."B=　"　B寄存器
它表示B寄存器中的内容。

④."DP=　　"　数据指针
等号后的4位是数据寄存器的地址指针，其左侧的高2位表示的是DPH，低2位表示的是DPL。

⑤."P0="…"P3="　　P0…P3口寄存器
它们用于查看和修改P0…P3口的内容。应该说明的是，由于P0口和P2口用作地址总线和数据总线，而总线上的数据是一直变化的，所以，窗口中的P0和P2的值是不确定的，或者说，它们的值是没有意义的。当使用POD8751仿真头时P0、P2口显示的是P0、P2口作为 I/O 口的值.

⑥."PSW=CASRSOXP" "BIN=00000000"　状态字寄存器
“PWS”表示状态字寄存器，等号右面的是8个状态位—C、A、F、R、S、0、X和P.
"BIN=00000000"等号右面的8位二进制数，与"PSW"中各位对应，表示它们的状态值。

8. 反汇编窗口
在CPU窗口的左下角部分是反汇编窗口，它是当前用户窗口中用户源程序经汇编后生成的目标码, 包括了地址码、机器码和反汇编程序。这个窗口提供的信息对于用户调试程序是十分有用的。

1.3.4.2 MCS96 系列 CPU 窗口

CPU窗口如图2.3.1所示，它包括下列的4个部分：

1. ADDR 0…F　（地址栏）

最上面一行标有"ADDR 0 1 …F "是地址名称栏，“ADDR”表示集中在一行中16个存储器(或寄存器)单元的第一个单元的地址，“0…F”表示它的下面对应着16个存储单元（或寄存器），“0”下面对应的2位16进制数，是第一个地址中的内容，“1”下面对应的是第二个地址单元中的内容…，“F”下面是第16个存储单元的内容。

2. ［程序存贮器］

它是为用户观察程序存贮器内部信息而设置的,共有64KB的地址空间,但是，在一般情况下只显示16个单元的信息。“ADDR”栏下的是以16进制表示的4位地址，地址后的16个数是这个地址及其随后的16个存储单元的内容。如果希望同时看到更多的信息，可用"↓"或"↑"键选择下一行的16个地址。若用鼠标，可单击右侧的激增框，将窗口放大到满屏幕，同时查看更多的内容。

当用鼠标选中某个地址单元后，直接按数字键或A…F键，将弹出一个窗口，用户可为该单元输入新的数值。

如果地址值较大，用鼠标翻页比较麻烦，可先用光标选中程序存贮窗口，再按ATL-G键，则可弹出一地址窗口，用户输入地址值后，光标将自动定位在要求的地址单元上，然后再修改这一单元的内容。

3. 工作寄存器区

CPU窗口的右下角是寄存器区，标有“PC=”…“SP=”、“P0=”、“P1=”等字样,它们可直接反映单片机中常用寄存器状态。它列出了15个寄存器的内容，分述如下：

①．”PC=” 当前 PC 值
②．"SP= " 当前堆栈指针值
③．"T0= "、 “T1= “ 计时器0、计时器1 值
④．"AD= " 模数转换器的结果
⑤．"AX="…"BX=" AX、BX、CX、DX 值, 其中 AX 从 1CH 开始
⑥．”’Z” 、 “N” 为 PSW 各位值
⑦．”INTM=“、 “INTP=“ 中断寄存器值
⑧. “WSR=“ 窗口选择寄存器值

4. 反汇编窗口

在CPU窗口的左下角部分是反汇编窗口，它是当前用户窗口中，用户源程序经汇编后生成的文件、包括了地址码、机器码和反汇编源程序。这个窗口提供的信息对于用户调试程序是十分有用的。

1.3.4.4 观察窗口
观察窗口标有“Watch"字样，如图2.4.1所示，用户可将一些程序中的变量和存储单元安排在观察窗口中，以便随时了解它们的变化情况。使用观察窗口观察变量前必须将欲观察的变量和存储单元设在窗口中。操作方法详见下面关于Debug菜单的有关操作。

1.3.4.5 用户文件编辑窗口
用户可以在该窗口中编辑、汇编、调试源程序。调试时，光标条所在位置，表明程序运行到此处。行列号表明编辑文件时光标所在的位置。应注意光标和光标条不同，光标是一短的闪烁的下划线。

1.3.5 窗口的操作方法
集成环境给已激活的活动窗口置以双线边界，活动窗口包括一个关闭框、标题框、窗口号、一个激增框、卷滚条和放大缩小角。编辑窗口中，当前行号和列号显示在窗口的左下角。如图2.3.7的用户文件编辑窗口。

1.3.5.1 打开窗口
启动MCS51软件后，屏幕上已打开的窗口是上次关机时的设置。如果没有所需的窗口，可按Windows\CPU 打开CPU窗口，用Windows\Watch打开观察窗口，用 File\Open 打开一个或数个用户文件编辑窗口。

1.3.5.2 激活窗口
在集成环境中已打开的每一个窗口都有一个窗口号，处于窗口边界的右上角。同时按下ALT和窗口号可以激活一个窗口（使之处于最前面），也可按F6依次逐个激活不同窗口，或在窗口任何可见部分单击鼠标左键。

1.3.5.3 放大缩小窗口
①. 放大到最大

激增框处于窗口右上角，在箭头处［↑］按下鼠标按钮，窗口扩展为最大尺寸。此时箭头[↑]变成双箭头［↑］，在双箭头［↑］处单击鼠标左键 ，将窗口恢复成以前大小。使用键盘，按 F5 即可实现窗口变最大或缩小。

②放大到要求的大小

在窗口右下角按住鼠标左键，拖动鼠标，窗口相应放大缩小。使用键盘设置窗口大小，要从WINDOW菜单中选择SIZE/MOVE 选项，或者按下CTRL-F5.使当前窗口变为单线窗口，再按Shift+箭头键,使窗口放大，最后用回车键结束放大。

1.3.5.4 移动窗口
窗口最上边是标题，在与标题平行的横线上按住鼠标左键，拖动鼠标，从而移动窗口。如果用键盘则应先激活欲移动的窗口，然后按Ctrl-F5,使该窗口的外框变为单线框，再按箭头键使窗口移动，最后用回车键结束移动。

1.3.5.5 关闭窗口
先激活欲关闭的窗口，用鼠标点关闭窗［‖］，或用Alt+-F3键关闭。

1.3.6 快捷键提示栏
1.3.6.1 快捷键提示栏的作用
快捷键提示栏位于屏幕最底下一行，它有如下作用：

1. 提示可用的基本热键。
2. 列出最常用的命令,使用户用鼠标或按键直接选择,避免在菜单栏中反复寻找.

1.3.6.2 快捷键提示栏的操作方法
1. 用鼠标选择快捷键提示栏中的某一命令项.
2. 用键盘操作时，按下快捷键提示栏提示的热键.

1.4 菜单及功能介绍

1.4.1 菜单的表示方法
为了便于叙述和查找各类菜单命令的用法，现将各菜单编号。编号的一般形式是“M\I”。其中用"\"号表示菜单命令上层与下层的关系，"\"号左边为上层，右边为下层。“M”是菜单名称,“I”是”M”菜单下的命令，例如，编号为“File\Open”的是主菜单中的第一项File菜单下的”Open”命令.依此类推。

各个菜单名之后若有圆括号，则圆括号中的是热键名，表示也可以用键盘直接完成该项命令，而不必通过菜单来层层选择。

1.4.2 选择菜单的方法
1.4.2.1 用键盘选择菜单命令
- 按下F10,激活主菜单栏.在主菜单栏中的各个菜单名里均有一个高亮字，也可用按Alt+高亮字的方法直接选中某项主菜单。
- 使用箭头键选择期望显示的菜单，然后按下ENTER键,弹出下层菜单后再继续选择.

1.4.3.2 用鼠标选择菜单命令

- 将鼠标置于所期望的菜单项上，按下鼠标按钮，选中此菜单。
- 如果弹出菜单后用户改变了想法，应则将鼠标移到菜单之外，这样不会选择任何命令。

1.4.3 单命令说明

1.4.3.1 FILE菜单 （ALT-F）

File\Open （F3） ［打开文件…］

打开一个编辑窗口,出现对话框，如图2.3.5所示，在输入框中输入要打开的文件名，或在列表框找到该文件后按下鼠标按钮，单击OK按钮表示打开一个新的编辑窗口；单击Replace按钮表示替代当前编辑窗口中的内容。

File\Save （F2） ［存盘］

将当前编辑窗口中的文件存到磁盘上。

File\Save as ［换名存盘］

更换文件名后再存盘。

File\Save OBJ as ［存目标码］

将目标代码存盘选此命令后将出现对话框。对话框中要求在”File Name”中输入文件名，在“Start address”和“ End Address”中填入始末地址。还应选择目标代码格式，共有4种：OBJ为 SICE格式机器码，HEX为16进制码，ROM是二进制文件格式(有些编程器称为BIN格式), RAM 格式同 ROM 格式, 但存盘的内容不是程序存贮器, 而是外部数据存贮器的内容 . MCS96 系列无RAM 格式。

File\Load OBJ as ［调入机器码］

将目标代码从磁盘上调入，操作与 File\Save OBJ as 相仿。

File\Change DIR ［改变目录］

指定新的当前驱动器和目录，仿真软件在当前目录中保存和查找文件。

File\DOS shell ［暂时返回DOS］

暂时退出集成环境，以便回到DOS环境中运行命令或其它程序，打入EXIT回到集成环境。应注意的是，此时仿真软件仍驻留在内存中，必须用“EXIT”命令返回集成环境，而不可再执行仿真软件的启动命令。

File\Exit (ALT-X) [退出]

退出集成环境，回到DOS环境中。在退出集成环境时，仿真软件将当前工作环境的一些设置参数保存在一个名为 *.DSK 的文件中。在下次进入集成环境时，仿真软件将工作环境自动恢复成上次退出时的状态。

File\About [关于…]

关于本仿真软件的版权说明。

1.4.3.2 Edit (ALT—E) [编辑]

Edit\Undo [恢复]

恢复本行修改前的状态。

Edit\Cut [裁剪]

删除程序中选定的正文，并将其置于裁剪板 Clip Board 窗口中。

Edit\Copy [复制]

保留选定的正文，同时将它拷贝到 Clip Board 窗口中，再用地 Edit\Paste 条命令将裁剪板中的内容复制在光标处。

Edit\Paste [粘贴]

将裁剪板中的内容插入到当前窗口的光标位置，裁剪板中的内容不变。利用裁剪板和本命令可实现文本块的移动和复制。

Edit\Show Clipboard [显示裁剪板中的内容]

打开 Clipboard 窗口，显示剪取的内容。

Edit\Clear [删除]

删除程序中选定的正文，但不将其置于裁剪板中。

1.4.3.3 Search 菜单 (ALT—S。) [查找]

Search\Find [查找…]

显示 Find 对话框，如图四所示，在输入框中打入欲查找的正文，设置影响查找方式的选项，然后单击OK按钮，开始查找。对话框中的参数选项有：

- Case sensitive: 区别大小写字母。
- Whole words only: 查找整个单词。

Search\Replace ［替换…］
显示一个对话框，在输入框中打入待查找的正文和用以替换的正文，设置影响查找和替换方式的选项，。命令参数选项有：

- Case sensitive: 区别大小写字母。
- Whole words only: 查找整个单词。
- Prompt on replace: 替换时提示用户。

Search\Search again ［再查找］
重复上一次 Find 或 Replace 命令操作。

Search\Find Label (ALT-G) ［查找标号］
显示一个对话框，在输入框中打入待查找的标号(程序地址)，然后单击OK按钮，开始查找。

1.4.3.4 Run 菜单 (ALT—R) ［运行程序］

Run\Program reset (CTRL-F2) ［程序复位］
中止当前调试过程，将CPU复位。如果是用 Run\Run 调试程序，则要用Ctrl+C，来中止正在运行的程序。

Run\Goto cursor (F4) ［执行到光标处］
程序从当前指针PC一直执行到光标所在的位置。如果光标所在的行不包含可执行语句，将显示警告信息：‘No code generate on this line’。

Run\Trace into (F7) ［跟踪执行］
跟踪执行程序。由用户控制一步一步地执行程序，碰到过程调用时仍将一步一步地执行程序中的每条语句，而不是一步执行完整个过程。

Run\Step over (F8) ［单步执行］
单步执行程序与跟踪执行相仿，不同的是当碰到过程调用时，一步执行完整个过程。

Run\Exec ［自动单步执行］
放慢程序执行的速度，以用户可以查看的速度一步一步的自动执行程序。

Run\Run（CTRL－F9） ［全速执行］
从当前 PC 处，全速执行程序。

Run\Set PC ［设置PC］
将程序PC值，设置为光标所在的行的PC值。

Run\Last PC ［寻上条指令］
将光带置于前一条指令所在行上，该命令没有改变当前的PC值。该命令只在软件模拟或已安装逻辑分析仪时有效.

Run\Next PC ［寻下条指令］
将光带置于下一条指令所在行上，该命令没有改变当前的PC值。该命令只在软件模拟或已安装逻辑分析仪时有效.

1.4.3.5 Assemble ［汇编］

Assemble\Assemble （ALT－F9） ［汇编］
汇编当前窗口中的文件。

Assemble\Make （F9） ［汇编主文件］
如果用户程序是由多个文件构成，其中有一个应为主文件，由它用 INCLUDE 伪指令引用其他的文件。按 F9 后，不论当前窗口是否为主文件，汇编总从主文件开始. 主文件由Option\Main File 命令设定。

Assemble\List ［生成列表文件］

汇编源程序并生成列表文件，后缀为 .LST。列表文件是一种同时包含有源程序、地址码、机器码的文件，对于调测，尤其是仿真器独立调测是必不可少的。

Assemble\Disassemble ［反汇编］
打开一个对话框，如图2.3.6所示，在File Name中填入文件名，在Start address， End Address中填入始末地址，按ENTER键开始反汇编。反汇编生成两种格式的文件：

- With Label ：带标号反汇编，缺省文件扩展名为.ASM。
- With address: 带地址，机器码反汇编,缺省文件扩展名为.DIS。

1.4.3.6 Compile 菜单 [编译]

见 38 页高级语言 Compile 菜单

1.4.3.7 Options [设置]

Option\Main file [设置主文件]

设置主文件。当需要将多个独立的文件汇编成为一个完整的文件时，应设置一个主文件来引用其他文件。执行 Assemble\Make 命令之前必须用这条命令。另外，一旦用本命令设置了主文件后，在执行 Assemble\List命令生成列表文件时，生成的是这里设置的主文件的列表文件。

Option\Project [项目设置]
Option\Compile [编译，连接控制项设置]

见 39 页高级语言 Option 菜单

Option\Debug options [设置调试窗口参数…]

选本项后可设置下列与用户源程序编辑窗口和源代码调试有关的参数：

- Create backup file :源文件存盘时自动建立后备文件。
- Insert mode :进入集成环境后处于插入方式。
- Autoindent mode :进入集成环境后源程序输入时处于自动缩排方式，即换行后 起始列不在第一列。
- New window :调试需要新文件时另外打开一个新窗口。
- Current window :调试需要新文件时使用当前窗口，当前窗口中的内容被新文件替代。

Option\ICE option [设置仿真器]

使用硬件仿真器要用到此命令。对于不同的仿真头, 设置有所不同。

MCS51 系列仿真器设置

根据所用的仿真器和仿真头，设置仿真时存储器所在位置和仿真头类型。在设置对话框中要求用户选择程序存储器（program memory)是用仿真器中的（Emulator），还是用户板上的(Target)。同时还要求用户选择仿真头(POD)是哪种类型的。

MCS96 系列仿真器设置

对于仿真器，每2K都可由用户选择这2K空间是在仿真器还是在用户板，[X] 表明2K在仿真器，[ ] 表明2K在用户板。 另外你还可以指定 CCR 中的每一位。

1.4.3.7 Debug （ALT－D） ［调试］

Debug\Add watch（ALT－F7） ［增加观察项…］

将新增观察项的表达式添加到观察窗口（WATCH)。选择此命令时，打开一个对话框，提示用户输入表达式和有关选项，定义观察变量的格式为：

［表达式］，［观察范围］［存贮器类型］［显示方式］

其中：

①．表达式

表达式可以是变量名，寄存器名，存储器名，寄存器地址，存储器地址，以及它们的运算式，如果是地址，必须在存储器类型选项中给出相应的参数（即指明是外部还是内部存储器）。

②．观察范围

如果在表达式中用户定义的是地址值或寄存器名，则可利用“观察范围”这个参数将相邻地址单元中的内容也列入观察范围，观察范围用数字表示，如“5”表示将相邻5个单元的内容也显示出来。

③．存贮器类型

用于指明表达式中定义的地址或寄存器是何种类型。类型分别用下列字母表示：

MCS51 系列

参 数	含 义	参 数	含 义
I	内部数据存贮器	P	程序存贮器
F	可位寻址存贮器	E	外部数据存贮器
缺 省	常量	X	8X52高128字节RAM

MCS96 系列

参 数	含 义	参 数	含 义
P	程序存贮器	缺 省	常 量
I	数据存贮器		

PIC 系列

参 数	含 义	参 数	含 义
I	内部数据存贮器	F	可位寻址存贮器
缺 省	常量		

对于高级语言,不需指定存贮器类型。因为高级语言在变量说明时已包含存贮类型。

④. 显示方式

它用于指定在观察窗口中显示变量或地址中内容时采用何种形式，可选下表所列字符：

参 数	含 义	参 数	含 义
H	十六进制	W	无符号字
D	十进制	SW	有符号字
B	二进制	S	有符号字节
缺省	无符号字节	C	字符

MCS51 系列设置观察变量举例：

0，8I　:从地址为0的内部寄存器开始，将连续8个寄存器的内容（即R0…R7），以十进制方式显示出来，其中省略了十进制参数D。

ACC,I　:以十进制方式显示A寄存器中的内容。

BUF，5HE :从标号为BUF(经汇编可知其地址)的地址开始，连续5个单元，以16进制方式显示外部数据存储器中的内容。

100+200H　:显示 100+200H 之和。

MCS96 系列设置观察变量举例：

1AH，8I　:从地址为 1AH 的内部寄存器开始，将连续8个寄存器的内容,以十进制方式显示出来，其中省略了十进制参数D。

1000H, I　:以十进制方式显示 1000H 号存贮单元中的内容。

100+200H　:显示 100+200H 之和。

Debug\Edit Watch　［修改观察变量］

编辑观察WATCH窗口中已设置了的观察项。

Debug\Delete Watch　［删除观察变量］

删除观察WATCH窗口中已设置了的某一个观察项。

Debug\Remove all watchs　［删除全部观察项］

从观察窗口中删除所有的观察项。

Debug\Evaluate and modify (CTRL－F4)　［求值/修改］

求表达式或变量的值，并让用户修改此值。选择此命令时，打开一个对话框，其中包括Expression(表达式)、Result(结果)、New Valuse(新值)三个域。用这个命令可以修改CPU中的各个内部RAM、特殊寄存器、标志位、程序存储器、外部数据RAM、程序中的变量等的数值。

- 在表达式栏中输入要修改的变量，按回车后，结果栏及新值栏中显示出变量当前值。
- 用 TAB 选择新值栏，输入修改后的值，按回车即可。

例:

- 在表达式栏中输入 ACC, I (这里的格式同 WATCH 项)，按回车，显示ACC当前值。
- 在新值栏中输入 10, ACC 的值就被改为 10。

Debug\Show elapsed time ［显示程序执行时间］

此命令可显示刚才执行的程序所用的时间。刚才执行的程序可以是一条、一段或全部。显示的程序执行时间，是以单片机晶体振荡器f=12MHz为基准。可用TRACE INTO(F7)统计一条指令的执行时间，也可用Goto cursor(F4)统计一段程序的执行时间。该命令只在软件模拟时有效。在硬件仿真时，可用逻辑分析仪统计程序执行时间。

Debug\Toggle breakpoint ［设置/取消断点］

在光标所在的位置设置断点，如原已有断点则取消该断点。

1.4.3.8 Window （ALT—W） ［窗口选择］

Window\CPU （ALT—1） ［CPU窗口］

选择CPU窗口。

Window\Watch （ALT—2） ［观察窗口］

选择WATCH窗口。

Window\Disassemble （ALT—M） ［反汇编窗口］

选择反汇编窗口。

Window\Register （ALT-R） ［内部寄存器窗口］

选择CPU窗口中右下角所示的各个寄存器。

Window\CODE RAM ［程序存储器］

选择CPU窗口中的程序存贮器。

Window\XDATA RAM ［外部数据存贮器］

选择CPU窗口中的外部数据存贮器。

Window\DATA RAM　　［内部数据存贮器］
选择CPU窗口中的内部数据存贮器。

Window\STACK RAM　　［堆栈］
选择堆栈段。

Window\BIT　　［位寄存器］
以位寻址方式选择位寄存器。

Window\Graphics　（ALT－F5）　　［图形仿真窗口］
选择图形仿真窗口，按任意键返回。只在软件模拟时有效。

Window\Size/Move（CTRL+F5）　　［窗口大小/位置］
修改活动窗口的大小和位置。用鼠标拖拉活动窗口的右下角可使窗口变大或缩小。用鼠标拖拉活动窗口与标题栏平行的双线，可使窗口移动。如果用键盘选SIZE/MOVE 选项后（或者按下CTRL-F5热键），使当前窗口变为单线窗口，再按箭头键,使窗口移动，最后用回车键结束移动。或按Shft+箭头键,使窗口放大，最后用回车键结束放大。

Window\Zoom　（F5）　　［最大/恢复］
将活动窗口放大成最大。若该窗口已经为最大，可选择此命令，将窗口恢复成原来的大小。
Window\Tile　　［平行排列多个窗口］
重新平行排列多个用户源程序编辑窗口。

Window\Cascade　　［重叠排列多个窗口］
重新重叠排列多个用户源程序编辑窗口。

Window\Next　　［选择下个窗口］
按窗口编号的顺序选择下一个窗口。

Window\Previous　　［选择上个窗口］
按窗口编号的顺序选择上一个窗口。

Window\Close　（ALT－F3）　　［关闭窗口］

关闭活动窗口。也可在窗口左上角的关闭框中按下鼠标按钮。

Window\List (ALT-0) ［显示窗口列表…］
列表窗口中显示的是用户当前打开的窗口。

Analyzers (ALT-Y) ［逻辑分析仪］
打开逻辑分析仪窗口。

1.5 速学实例

1.5.1 进入汇编集成调试环境

在相应的子目录中打入MCS51(对于MCS96打入MCS96)即可。

C>MCS51 (对于MCS96打入MCS96)
进入汇编集成调试环境时可用选项:

/S: 使用硬件仿真器
/1: 使用串口 1
/2: 使用串口 2
/D: 降低通信速率
通常情况下，你不用选择波特率，仿真软件将自动地选择最合适的波特率进行通信。只有你在使用过程中，经常出现通信错误时，才需人工调整。
/I: 使用INST区分程序/数据存储器。(仅用于MCS96系列)

例:

C>MCS51
使用软件模拟，不需使用硬件仿真器。

C>MCS51/S2
使用硬件仿真，用串口2进行通信。

1.5.2 调试程序

(1) 输入程序: 按 F3 键，按要求输入文件名及ENTER。

F3

FIRST　　这里设要输入的文件名为 FIRST
ENTER

这时系统打开一个名为FIRST的编辑窗口，你可以在编辑窗口中输入汇编程序。

MCS51 系列　　　　　　　　MCS96 系列

```
       Index equ   20h          Index equ   20h
       Sum   equ   21h          Sum   equ   21h

       org   0000h              org   2080h
Start:                     Start:
       mov   Index, #5          ldb   Index, #5
       mov   a, #0              ldb   Sum, 0
Loop:                      Loop:
       add   a, Index           addb  Sum, Index
       djnz  Index, Loop        djnz  Index, Loop

       mov   Sum, a             jmp   Start
       jmp   Start

       end                      end
```

这个程序的功能是计算 1+2+3+4+5，结果存于变量 Sum 中，Index 为循环控制变量。

(2) 保存程序，按 F2 即可

```
F2
```

(3) 汇编程序，按 F9 即可

```
F9
```

集成环境首先对源程序进行汇编,若发现错误,立即返回编辑状态，并打开一个错误信息窗口,用户可按提示修改源程序，若无错误则返回编辑窗口等待调试。

(4) 观察变量，如想观察变量可按如下操作:

```
CTRL-F7
```

```
Index, I        观察 Index
CTRL-F7
Sum, I          观察 Sum
```

(5) 单步执行程序

按 F7 一条一条的执行下去,注意观察窗口中 Index, Sum 的变化情况, 源程序中的加亮条表示下一条将要执行的指令。

```
F7  F7  F7  ...
```

(6) 软件复位 CPU

按 CTRL-F2 复位 CPU

```
CTRL-F2
```

(7) 断点执行程序

要使程序执行到某条指令处暂停, 如希望程序执行到 jmp Start 处暂停可按如下操作:

```
将光标移至 jmp  Start 所在行
F4
```

(8) 全速执行程序

```
CTRL-F9
```

(9) 中断执行程序

如果使用 - 硬断点仿真器, 打入 CTRL-C 即可返回监控。

如果使用 - 软断点仿真器, 则必须按仿真器上的 RESET 键进行硬件复位。

1.5.3 显示与修改内存单元

在软件调试过程中往往需要根据软件运行的情况及时修改CPU内部RAM、特殊寄存器、有关的输入/输出口、外部数据RAM等。方法和步骤是:

1.5.3.1 使用 CPU 窗口显示与修改内存单元

①.选择窗口

用鼠标选择CPU窗口中的有关窗口。或按下列键选择窗口

- ALT-T: 选择寄存器
- ALT-P: 选择程序存贮器
- ALT-M: 选择反汇编
- ALT-N: 选择外部数据存贮器 (仅对MCS51)

- ALT-I: 选择内部寄存器　　(仅对MCS51)
- ALT-K: 选择堆栈　　(仅对MCS51)
- ALT-L: 选择位存贮器　　(仅对MCS51)

②.选择指定单元的地址
使用ATL-G键，在弹出的地址窗口中输入欲修改单元的地址值，回车后光标定位在指定地址单元上。或使用上,下,左,右,PageUp,PageDown 键选择单元。

③.修改
当光标定位在要求的单元上后，直接输入数据（0…1，A…F），此时将弹出一修改窗口，可在窗口的“NEW VALUE”栏输入欲修改的数据，数据格式可以是10进制、16进制、2进制或表达式。

1.5.3.1 使用观察窗口显示与修改内存单元
可利用设定观察项的方法，将欲观察的寄存器、存储器或程序中的某变量放入观察窗口中。

①.按 Debug\Watch 所述方法设定观察项。
②.修改变量：按CTRL－F4打开显示修改窗口，在EXPRESSION中输入观察项（观察项定义同上述WATCH一样），用TAB键选择NEW项，输入新值并按ENTER。

1.5.4 中断用户的程序运行
如果使用软件模拟或使用硬件仿真，执行”RUN”全速执行的命令后，打入”CTRL-C”可中断程序的运行，光带停留在中断处，在中断处进行观察修改等操作后，再按“RUN”可从中断处继续往下执行。如果使用-仿真器，则需按仿真器上的[RST]键，中断程序的运行，使程序返回0000地址处。这里所说的中断是指用户出于调试的目的，人为地使程序停下来，与程序中的中断处理是两回事。

1.5.5 中止用户程序的运行
当执行了运行命令后, 打入”CTRL-F2”可中止程序的运行，其作用相当于按“RESET”键，使整机复位，程序指针返回0000地址处。如果程序正在运行, 应先用2.5.4 条中断程序运行。

1.6 WAVE汇编器

1.6.1 表达式

表达式	功能	例
HIGH(DATA)	取字的高字节	HIGH(1234H) = 12H
LOW(DATA)	取字的低字节	LOW(1234H) = 34H
MASK(BIT)	取位屏敝字	MASK(20H.1) = 00000010B
!	取反	!00001111B = 11110000B
&	与	0FFH & 07H=07H
*	乘	3*4 = 12
/	整除	5/2 = 2
%	取模	5%2 = 1
<	左移	111B<3 = 111000B
>	右移	111B>1 = 11B
^	异或	101B^111B = 010B
+	加	10H+20H = 30H
-	减	20-10 = 10
|	或	010B|101B = 111B
$	当前指令的首地址	$+100 = 当前地址+100
.	位地址运算	20H.1 = 1

1.6.2 CALL, JMP 指令

这两条指令并非新的指令,汇编程序最终将生成SJMP,AJMP,LJMP,ACALL,LCALL中的一条指令,汇编器自动选择指令码较短的一条指令。

1.6.3 过程(PROC)伪指令

格式为:

```
      Proc  Proc1, Proc2, ... , Proc30
Proc1:   ...
    .
Proc2:   ...
    .
    .
Proc30:  ...
```

```
.
  END
```

一个过程的定义由 PROC 开头，在 PROC 后面最多可定义30个公有标号，过程用 END 结束。除了定义为公有标号的那些标号，其余标号只在过程中有效，过程结束即消失。例：

```
        proc  Sub1
Sub1:
        mov   a, #0
        mov   r0, #10
Loop:
        add   a, r0
        djnz  r0, Loop
        ret
        end

        proc  Sub2, Sub3
Sub2:   mov   B, #10
Sub3:
        mov   a, #1
        mov   r0, #10
Loop:
        mul   a, B
        djnz  r0, Loop
        ret
        end
```

上述两个过程中共使用了四个标号: Sub1, Sub2, Sub3, Loop。其中 Sub1, Sub2, Sub3 是公有的，可在程序的任意处使用。而Loop为私有标号，只在定义它的过程中有效。因此两个过程中的 Loop 是不相同的标号。另外编程时还应**注意：PROC和END必须成对使用，否则编译时将提示出错。**

1.6.4 包含文件 (INCLUDE)伪指令

包含功能允许你将几个文件串起来汇编。

格式：INCLUDE "FileName"

汇编程序用 FileName 文件的整个内容来替换INCLUDE 语句。

如有三个独立的文件

S1.ASM　　　S2.ASM　　　M.ASM

```
MOV A, #1           MOV  B, #22        CLR C
INC A               DEC  B             INCLUDE "S1.ASM"
                                       INCLUDE "S2.ASM"
                                       ADD A, B
```

汇编 M.ASM 相当于汇编如下一个文件:

```
CLR C
MOV A, #1
INC A
MOV B, #2
DEC B
ADD A, B
END
```

1.6.5 IF ELSE ENDIF 伪指令

格式为:

```
IF  表达式1  =表达式2
  .
  .
ELSE
  .
  .
ENDIF
```

如果表达式1的值与表达式2的值相等,则汇编IF与ELSE之间的源程序，否则汇编ELSE 与ENDIF 之间的源程序。可以没有 ELSE 分枝。

1.6.6 汇编错误信息

01: Bad operation

操作码错

02: Bad operand

操作数错

03: Bad constant

常数错

04: Bad string

字符串错

05: Repeat define or constant

重复定义或常数.汇编程序期望一个标识符，但却出现一个常数,

或虽是一个标识符，但它已被定义过

06: Divide by zero

被 0 除 ，表达式中 0 作为除数

07: Invalid character

非法字符，该字符在汇编语言中无定义

08: Null string

字符串长度为 0

09: UnExpected "END" in include file

在包含文件中出现了不期望的 "END",一个程序不能在包含文件中结束.

10: ")" Expected

期望 ")"

11: "(" Expected

期望 "("

12: "CR" Expected

期望 "CR"

汇编程序已成功的汇编了一行，但这行中还有内容未用到.

14: "," Expected

期望 ","

15: Procedure name error

过程名应是一个未定义的标识符.汇编程序期望一个标识符，但却出现一个常数，或虽是一个标识符，但它已被定义过

16: PROC symbol not define:

过程已结束，但还有过程入口没有定义

17: File name Expected

期望文件名. INCLUDE, USES 命令行中需要文件名

18: Bad ORG

ORG 定义的地址错

19: Jump out of range

跳转出范围

20: Too many FORWARD define

太多的向前定义 (大于 2K 个)

21: Too many labels in program

程序中标号太多(>1K). 程序中的标号太多，建议使用汇编程序提供的过程结构.

22: Too many labels in procedure

过程中标号太多(>1K). 一个过程中的标号太多,一个过程不应有这样大,你可将这个过程分成若干个规模小点的过程.

23: Too many debug file

被调试的文件太多，一个程序最多可调试含 48 个子文件的源程序.

24: Bad bit address

位地址错. 位地址的范围是 ：

1. 20H - 2FH
2. 大于 80H 且能被 8 整除.

25: PROC can not in procedure

过程中不能定义过程. 过程是不可嵌套的

26: Too many Entry in procedure

过程的入口太多. 一个过程最多可有 30 个入口. 如果你的过程入口多于30个请分成若干个规模小点的过程

28: Symbol not define:

标号未定义

29: Program out of range

程序超出了程序存贮器的地扯空间

30: DB or DW too long

DB 或 DW 太长. 每个 DB 和 DW 最多允许定义 255 字节,
你可用几个 DB 或 DW 来定义大量数据

31: Too many lines

源程序行太多列表及源代码调试功能允许源程序的最大行数为 96K 行.

32: UnExpected "END" in Procedure

过程中不期望的 "END". 过程已结束但还有入口没有定义

33: Too many labels

在列表时标号太多

34: PROC not end in unit

单元中的过程必须在单元中结束

35: "END" Expected

期望 "END"

36: TITLE String Expected

期望标题

TITLE 语句中必须有一标题字符串，如果不想打印标题可设为一空白字符串,
TITLE 的初值即为一空白串.

37: String too long

字符串太长. 字符串的长度最大为 255 字节

38: IF too deep

IF 嵌套太深，最多可嵌套8层

39: File too large

汇编源程序文件太大，一个文件最大为64K,可用INCLUDE语句将源程序放置在几个文件中.

40: "]" expected

期望 "]"

41: "[" expected

期望 "["

50: IF expect

期望 "IF"

51: ENDIF expect

期望 "ENDIF"

52: "=" expect

期望 "="

Internal error ?? - Please report to CHY

其中 ?? 为内部错误号， 请将这一错误告知:

邮编: 210018
南京市 珠江路222号 长发大厦5楼B座　　陈小宇
电话:(025) 3192459、3193973
Email: wave-cn@263.net
网站：http://www.wave-cn.com

1.7 软件模拟器

1.7.1 定时、记数器

软件模拟的定时/记数器时序与真实的MCS51定时/记数器完全一致。当使用记数器时，所需要的记数脉冲由主机键盘提供，方法为：在程序运行时（菜单条右端为Running时）：

按下	功能
F5	T0脉冲输入脚置为低电平
F6	T0脉冲输入脚置为高电平
F7	T1脉冲输入脚置为高电平
F8	T1脉冲输入脚置为低电平

如按下按键F5, 再按F6将在T0脉冲输入脚产生一负跳变，使记数器加1。

1.7.2 中断系统

软件模拟的中断系统与真实的MCS51中断系统完全一致，当中断来自外部时用主机键盘模拟，方法为：在程序运行时（菜单条右端为Running时）：

按下	功能
F1	INT0脚置为高电平
F2	INT0脚置为低电平
F3	INT1脚置为高电平
F4	INT1脚置为低电平

如按下按键F1, 再按F2将在 INT0 脚产生一负跳变，引起中断。

1.7.3 串行口系统

软件模拟的串行口系统与真实的MCS51串口完全一致。接收时，串行脉冲用主机键盘模拟，方法为：在程序运行时（菜单条右端为Running时）按下F9。此时弹出一对话窗口，在此输入0,1脉冲信号，高位在前,0表示低电平，1表示高电平，要包括起始位、停止位和校验位 (如果有)。当按下 ENTER 后,相当于将该0,1串所表示的脉冲信号加到 MCS51 的串行输入脚上。波特率自动定为当时设定的接受波特率。

1.8 高级语言调试环境

本节仅涉及软件的**高级语言**集成调试环境的使用, 有关**高级语言**本身请参考相关语言手册及资料。

1.8.1 安装高级语言集成调试环境

见 1.2 节

1.8.2 集成调试环境对使用的限制

1. PL/M 源程序(包含文件除外)必须用 .PLM作为后缀。汇编程序必须用.ASM 作为后缀。C 源程序(包含文件除外)必须用 .C作为后缀。
2. PL/M 文件名必须与模块名相同。

1.8.3 进入高级语言集成调试环境

在相应的子目录中打入　即可

C>ICES

进入　时可用选项

/S: 使用硬件仿真器

/1: 使用串口 1

/2: 使用串口 2

/D: 降低通信速率

　　通常情况下，你不用选择波特率，ICES软件将自动地选择最合适的波特率进行通信。只有你在使用过程中，经常出现通信错误时，才需人工设置。

/I: 使用INST区分程序/数据存储器。（仅用于MCS96系列）

例:

```
C>
```

使用软件模拟，不需使用硬件仿真器

```
C>/S2
```

使用硬件仿真，用串口2进行通信

1.8.4 调试程序

(1) 按F3键，要求输入文件名，打入文件名FIRST，按下 ENTER 键后便可打入下列程序。

```
F3
FIRST      这里设要输入的文件为 FIRST
ENTER
```

```
First: do;      /* 这里的模块名First与文件名First.PLM相同 */
  declare I byte;
  declare J byte;

  J = 0;
  do I = 1 to 10;
    J = J+I;
  end;
end First;
```

这个程序的功能是计算 1+2+3+4+5+6+7+8+9+10,结果存于变量 J 中。

(2) 保存程序，按 F2 即可。

```
F2
```

(3) 编译程序，按 F9 即可。

```
F9
```

集成环境首先对源程序进行编译,若发现错误,立即返回编辑状态,并打开一个名为ERR.MSG的错误信息窗口，用户可按提示修改源程序,若无错误则自动进行连接后返回编辑窗口等待调试。

（4）观察变量

按 CTRL-F7，打入I并按 ENTER,此时 I 显示在 2 号观察窗口。用同样的方法输入 J。

CTRL-F7
I 观察 I
CTRL-F7
J 观察 J

（5）运行程序

按 F7 一条一条的执行下去,注意观察口中 I,J 的变化情况。源程序中的加亮条表示下一条将要执行的指令。

F7 F7 F7 ...

1.8.5 有关 ICES 菜单

1.8.5.1 Compile\Compile

编译当前窗口文件

1.8.5.2 Compile\Make

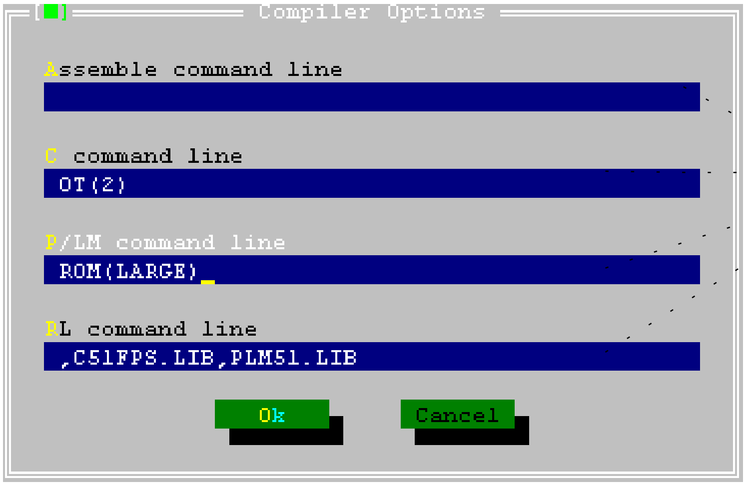
按项目表进行编译，若源文件在上次编译后未做修改，则不再编译该文件。

1.8.5.3 Compile\Build

按项目表进行编译，无论源文件在上次编译后有没有做修改，编译所有源文件。

1.8.5.4 Option\Compile

设置编译项及连接项。



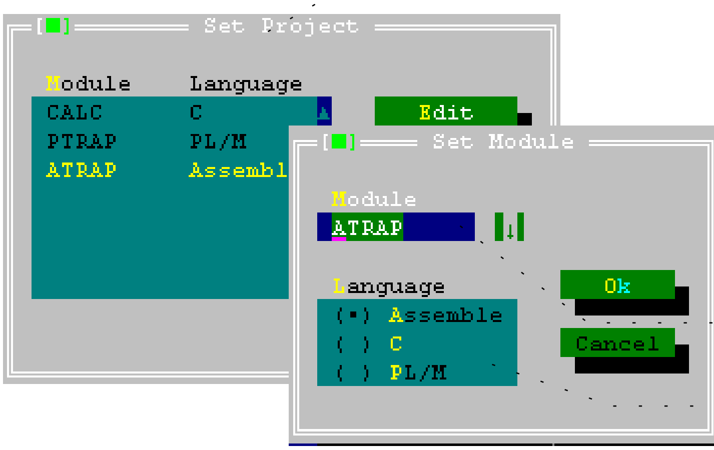
汇编命令行
C 编译命令行
PL/M 编译命令行
连接命令行

集成环境在编译命令行中自动的加入 DB、SB、TY 编译控制项，用于源码调试。用户不可在编译命令行中再次定义

1.8.5.5 Option\Project

设置项目表。一个大型程序可由多个模块组成,这些模块可以是用PL/M，C高级语言编写的，也可以是用汇编语言编写的。其中第一个模块为主模块。

模块名	文件名	语言
CALC	CALC.C	C 语言
PTRAP	PTRAP.PL M	PL/M 语言
ATRAP	ATRAP.AS	汇编语言



在这里输入模块名

这里选择语言

除了以上的几个命令，其余命令与汇编调试软件相同，请参阅汇编调试软件说明。

二 用户板硬件测试

仿真器具有先进的硬件测试功能。可以将动态执行的指令静态化，配合逻辑笔或电压表，可很容易地查出各种硬件连线及逻辑错误。

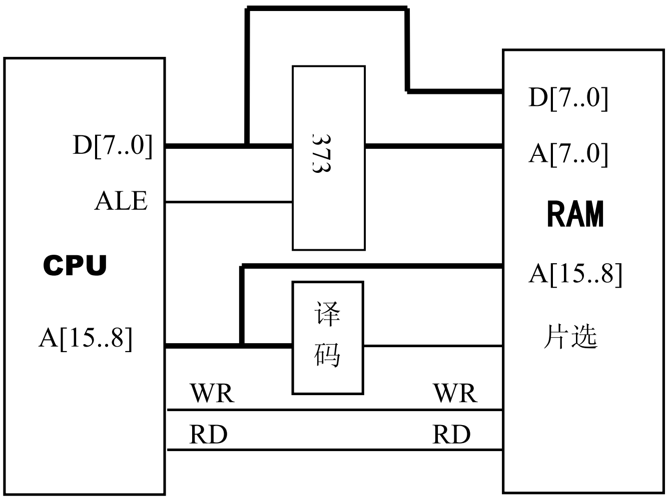
2.1 启动硬件测试功能

在DOS提示符下打入TEST 回车，TEST.EXE被解压。

C>ICESTEST/S1 或 S2。即可进行硬件测试。

2.2 应用举例

测试如下存储器系统



测试 1234H 单元，写入 56H，再读出。

1. 测试D[7..0]

   按D[7..0]按钮，输入34H，按OUTPUT按钮。此时34H将输出在D[7..0]端口。

   测试373输入端及RAM的数据端，应为34H。

2. 测试地址锁存

按ALE按钮，ALE输出1，此时373输出端及RAM的低位地址端应为34H
再次按ALE按钮，ALE输出0，此时34H被锁存在373输出端口。

3. 测试A[15..8]
按A[15..8]按钮，输入12H，按OUTPUT按钮。此时12H将输出在A[15..8]端口。
测试RAM的高位地址端，应为12H。测试译码电路，此时RAM的片选端应有效。

4. 测试写信号
按D[7..0]按钮，输入56H，按OUTPUT按钮。此时56H将输出在D[7..0]端口。
测试RAM的数据端，应为56H。此时373输出端应保持为34H。
按WR按钮，WR输出0。再次按WR按钮，WR输出1，此时56H被写入1234H单元。

5. 测试读信号
按RD按钮，RD输出0。此时，RAM的数据端应为56H，同时显示器上的D[7..0]应显示56H。

三 问与答

1. 为什么串口通信出错?

串口通信出错有几种可能:

1. 用户板有问题, 你可以将仿真头从用户板上取下,将晶振跳接至仿真头上。测试不连用户板时仿真器串口通信是否正常。
2. 串口电缆连接有问题。通信时**仿真器**上的指示灯应闪烁， 若不闪烁, 说明通信电缆连接有问题或串口号选择不对。
3. 串口号和波特率选择不对。一般 PC 机上有两个串口:串口1及串口2,在调试程序时应正确选择串口: /S1选择串口1, /S2选择串口2。另外波特率选择不对也会引起通信出错。一般情况下你可以不选择波特率, 由软件自动选择一个合适的波特率, 但是对有些机器可能选择得不合适, 这时你可以用几个波特率试试, 手工选择一个合适的波特率。如可以输入 /S2D 选择串口2, 降低通信速率。

2. 为什么程序工作正常但用户板上的外设却不工作?

这通常是你没有正确设置仿真存贮器,见OPTION/ICE设置。

3. 问： 如何选择 G6 系列仿真器的 51 仿真头?

答：使用 G6 系列仿真器来仿真 8951CPU 时，应根据 P0 口和 P2 口的使用方法来选择仿真头，如果用户要求 P0 口和 P2 口作为总线来用，应选拔 8051 仿真头；如果用户要求 P0 口和 P2 口作为 I/O 来用，应选择 89C52 仿真头。

4. 问：如何将程序写入芯片?

答：DOS 环境写 89C5X、写 27XXX 将已经调试正确的源文件编译一次，输入

ALT——F (File）V (Save OBJ as）

即进入“Save object file”界面，设入文件名（如 DEMO）起始地址，结束地址， 选用 ROM 格式,然后按回车键或“V”,在当前目录里会产生一个后缀为 ROM 的文件。如 DEMO。ROM。

打开并进入编程器环境，将 DEMO ROM 文件以二进制的形式调入到编程器内，然后再写入 89C5X 或者 27XXX。

5. 问：在 DOS 环境下如何调入较大的源程序文件?

答：在 DOS 环境下，因受系统机显示内存的限制，调入较大的源程序时，常会出现内存不够的警告，为了避免这种现象，伟福编辑器采用了一种一主带多从的结构模式，具体做法

如下：

（1）将一个大文件分成若干个小文件，每个小文件的源程序行数控制在 1000 行左右，最多不要超过 1500 行。

（2）选择其中一个文件作为主文件，在主文件中用 INCLUDE 伪指令包含其余的若干个文件。

（3）将主文件名输入到 Option 中的“main　File”中去。这样就可以调入，编译了。这种结构同样适用于 80196 系列和 PIC 系列等。本系统提供了一个例子 JFQmain.ASM，供您参考。

另外，如果 MCS.DSK 文件被破坏，也会出现内存不够的现象，这时需要删除掉 MCS.DSK 文件后，重新进行设置。

6. 问：仿真调试时“复位”是怎么回事？

答：如果仿真系统要求“复位”，实际上是仿真系统无法工作，出现这种现象的原因有很多，如晶振跳线位置不正确，用户板上无电源，仿真头接触不良，仿真头的选择和仿真系统内的设置或选项不一致等。如果在使用中出现这种情况，可以先将仿真头从用户板上拔下来，把晶振跳线器放在“1”的位置上，再次进入仿真系统。如果正常，则问题在用户板上，否则问题在仿真器上。

有看门狗的芯片（如 PIC、80196、LPC764 等）在仿真时应关闭看门狗。否则也有可能导致“复位”。

7. 问：仿真时工作正常，程序固化后却不能运行？

答：出现这种情况有下列可能性，供参考：

（1）　用户板的程序片电路有问题：如 PSEN 不通或者接错。电路插座是 2764，但程序片用 27128 或 256，地址线 A14 浮空等。使用 89051 内部的程序，但 EA 未接高电平等。

（2）　用户板晶体振荡电路有问题，用户板复位电路有问题。

（3）　用户系统内有需要复位的接口电路，如 8155、8255、8279 等。在它们还未完成复位时，如果 CPU 就给他们写控制字，可能会造成它们工作的不正常。在这种情况下，CPU 复位后应延时一段时间（如 100ms），以确保 8155 等已被完全复位，然后再向 8155 等写控制字。如果用户系列内有智能 MODE 等外设，往往还需要更多的延时，才能确保工作正常。

（4）　堆栈溢出

8051 系列的单片机有的有 256 个 RAM，有的只有 128 个 RAM，现在高级语言编译系统都默认为 256 个 RAM。如果您使用只有 128 个 RAM 的芯片，就应注意堆栈是否会溢出，否则有可能不能运行。试验的办法也简单，用一片 256 个 RAM 的芯片一试就知道了。

（5） 程序片烧写不正确，程序片烧写不正确的情况有好多种：

A. 格式不对或者内容不对。

B. 光写程序未用芯片的设置字，如 PIC、LPA764 、80C196 的保留字等。这些设置字中凡是需要用户写的应正确按需要填写，不需要用凡写的应写入“FF”（不写）。

（6） 检验程序片烧写正确与否的方法是，从一烧好的片子中读出机器码，并行成机器码文件，然后让仿真器运行这个机器码文件。如果运行正确，可以排就除这个因素，否则就应重新按正确的方法写片子。系统接线有错，特别是 80196 系列芯片，如果 READY 脚、NMI 脚浮空，则也会出现脱机后不能运行的现象。

8．问：调试时为什么不能在源程序上进行？

答：如果源程序文件编译不正确或者链接出错，仿真时就会跳到 CPU 窗口，出现这种现象，应仔细检查编译和链接是否正确。另外，在 PLM 语言中，模块文件的头、尾，文件名应相同，比如说，某一模块文件名为 ABC.PLM，则该文件的头应有 ABC：d0；结尾应该是 END ABC;，否则也会出现这种现象。

9. 在汇编时出现标号太多怎么办?

调试软件中的汇编器（DOS版）支持1024个标号，当标号多于1024时就会发生这种错误。你可以使用调试软件引入过程伪指令来解决这个问题。使用过程伪指令可使用多达 1M 的标号，且允许同名。详见汇编器一章。